République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

UNIVERSITE SAIDA - Dr. MOULAY Tahar

FACULTE : TECHNOLOGIE

DEPARTEMENT : INFORMATIQUE

MEMOIRE DE MASTER

OPTION :

SECURITE INFORMATIQUE ET CRYPTOGRAPHIE

Thème

Détection et filtrage de courriels indésirables par les techniques Bio-inspiré

Présenté par :

Medani Tarek Houssam Eddin

Belkhir Abdeldjalil

Encadré par :

Mr : Latreche Abdelkrim

Promotion :
2019-2020

REMERCIEMENT

Avant toute chose nous remercions Allah le tout puissant de nous avoir accordé la force et les moyens afin de pouvoir réaliser ce travail.

Au terme de ce travail nous adressons tout d'abord nos sincères remerciements à : Mr. Latreche Abdelkrim pour avoir dirigé ce travail et accepté d'encadrer, pour ses conseils st ses orientations

Au Département des

Science et technologie, pour avoir accepté d'examiner ce travail.

A tous les enseignants de Départements de technologie.

DÉDICACES

Mes parents à qui je dois ce que je suis longue vie

Mes très chères et Mes très chères

A toute ma famille grands et petits

Toute la famille : Belkhir et Medani

A tous mes ami(es) un par avec qui j'ai passé de merveilleux

moments et avec qui j'ai partagé malheur et bonheur

Tous mes enseignants

Toute la promotion de « Sécurité Informatique Et

Cryptographie »

Table des matières

Chapitre 5 : Contribution

Liste des figures

Liste des tableaux

Liste des algorithmes

Résumé

Parmi les problèmes qui affectent négativement la communauté scientifique en particulier, et les internautes en général, figure le problème de la sécurité des données. Pour cela, et dans le cadre de ce mémoire, nous avons essayé d'utiliser des algorithmes inspirés de la vie naturelle et des sociétés de certaines créatures qui vivent dans un monde organisé.

Les résultats obtenus dans la détection du SPAM (qui est notre sujet) sont comparés aux résultats obtenus à l'aide d'algorithmes précédemment utilisés

الملخص

من بين المشاكل التي تؤثر بشكل سلبي على المجتمع العلمي خاصة ومستخدمو الانترنيت عامة هو مشكل امن البيانات. لهذا الغرض وضمن هذه الاطروحة حاولنا استخدام خوارزميات مستوحات من الحياة الطبيعية ومن مجتمعات بعض المخلوقات التي تعيش ضمن عالم منظم

وتتم مقارنة نتائج المتوصل لها في الكشف عن الرسائل الاقتحامية (الذي هو موضوعنا) مع نتائج متوصل لها باستخدام خوارزميات مستعملة مسبقا.

Abstract

Among the problems that negatively affect the scientific community in particular, and Internet users in general, is the problem of data security. For this purpose, and within this thesis, we tried to use algorithms inspired by natural life and from the societies of some creatures that live within an organized world.

The results of the results of the detection of SPAM (which is our topic) are compared with the results obtained using previously used algorithms.

Introduction général:

Le courrier électronique (ou courriel) est aujourd'hui l'une des applications les plus utilisées sur internet et sur les réseaux d'entreprises. Utilisé pour des applications très variées (personnelles, professionnelles, associatives, etc.) celui-ci tend à prendre une place de plus en plus importante par rapport aux moyens de communication traditionnels. Outre son faible coût, la messagerie électronique a l'avantage d'optimiser la communication et la diffusion d'informations. Il est impossible de donner une liste exhaustive de ces avantages, mais il est évident que le courrier électronique:

- Permet une économie de temps et de moyens.
- C'est un moyen de communication rapide et relativement moins cher (comparé au courrier par avion ou au fax),
- Il permet d'envoyer un message à plusieurs destinataires simultanément,
- Et échanger des messages à n'importe quelle heure, en dépit des différences des fuseaux horaires,
- Et enfin, il permet de transmettre des documents de données audio et vidéo, etc.

Cependant ces dernières années, l'utilisation des courriers électroniques a conduit à une nouvelle escalade de problèmes causés par le volume des messages non sollicités connus sous le nom de spam. Le problème des courriers électroniques non désirés est aujourd'hui un problème sérieux. L'agence européenne ENISA* (Agence Européenne de la Sécurité des Réseaux et de l'Information) vient de sortir une nouvelle étude selon laquelle 95,6% des messages électroniques seraient identifiés comme étant des spam par les chaînes de filtrages des fournisseurs d'adresses email. Les conséquences du spam aussi bien sur le plan individuel que dans les entreprises sont significatives; elles peuvent être catastrophiques pour les entreprises qui ne sont pas préparées pour faire face à ces menaces. Le spam n'est plus simplement ennuyeux ; il est coûteux pour les entreprises non seulement financièrement, mais également en termes de temps de traitement, d'utilisation de bande passante, de gestion et de consommation de ressources.

Pour faire face à cette charge croissante de spam, de nombreuses solutions ont été proposées. Certaines solutions sont basées sur l'en-tête du courriel et utilisent les listes noires, les listes blanches, la vérification de DNS pour détecter le spam. D'autres solutions comme le filtrage à base d'apprentissage automatique, se basent sur le contenu textuel du courriel. Dans cette étude, nous intéressons à l'application de l'apprentissage supervisé (AS) et deux méthodes bio-inspiré {SWBs

(social workes bees) et ASR (artificial social roachs)} pour la détection de spam. Les solutions existantes déjà d'atteindre une très grande exactitude. Cependant, les quantités énormes de spam diffusées aujourd'hui nous encouragent à améliorer encore ces solutions ou proposer de nouvelles solutions pour atteindre une meilleure qualité de détection.

Chapitre 1

1- Catégorisation de textes

1.1- Introduction :

Le problème de filtrage des courriels indésirables est abordé dans ce mémoire comme un problème de catégorisation de textes à deux catégories : la catégorie spam pour les courriels indésirables, et la catégorie légitime pour les autres courriels légitimes.

Il faut donc disposer d'un ensemble d'exemples pour chaque catégorie, préalablement étiquetés. Et grâce à ces deux ensembles de courriels, il est possible de construire un classifier avec un algorithme d'apprentissage supervisé. Si ce dernier est correctement conçu, il sera capable de prédire pour chaque nouveau courriel sa propre catégorie.

Jusqu'aux années 1980, l'approche dominante en CT était celle de l'ingénierie des connaissances. Il s'agissait de modéliser, sous forme de règles, les connaissances d'un expert sur les règles de classification des documents. Ces règles étaient utilisées par le système pour déduire la classe d'un nouveau document.

1.2- Catégorisation de textes :

La tâche de catégorisation de textes (appelée aussi classification de textes) est une tâche ancienne de la recherche d'information (Manning, et al. 1999) qui est apparue au début des années 60 mais qui s'est largement développée durant les 15 dernières années. Elle consiste à attribuer à un document une ou plusieurs catégories (ou classes) parmi un ensemble prédéfini.

Cette problématique a dernièrement trouvé de nouvelles applications dans des domaines tels que le filtrage de spam, le routage des courriels (voir Figure 1.1), la veille technologique, l'analyse des opinions, la détection des intrusions, etc.

Dans cette section, nous nous intéressons tout d'abord à la définition de cette tâche, puis nous décrivons brièvement le processus d'un système de catégorisation de textes.

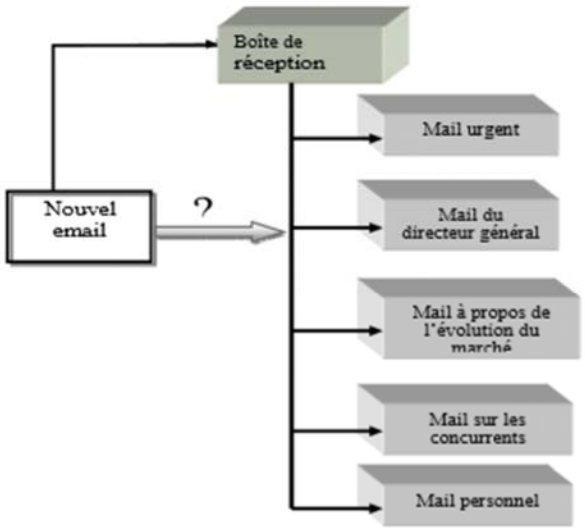

Figure 1.1-Un exemple d'un système de routage de courriels.

1.3- Définition :

La catégorisation de texte consiste à chercher une liaison fonctionnelle entre un ensemble de textes et un ensemble de catégories (étiquettes ou classes). Cette liaison fonctionnelle que l'on appelle aussi modèle de prédiction est considérée par un apprentissage automatique. Pour ce faire, il est nécessaire de disposer d'un ensemble de textes préalablement étiquetés, appelé ensemble d'apprentissage, à partir duquel nous estimons les paramètres du modèle de prédiction le plus performant possible; et qui produit le moins d'erreur de prédiction (Sebastiam, 2002).

Nous pouvons distinguer trois types de catégorisation de textes :

- **Catégorisation binaire** : ce type de catégorisation correspond au filtrage. Elle permet, par exemple, de répondre aux questions suivantes : « le document est pertinent ou non? », « le courriel est un spam ou non »?

- **Catégorisation multi catégories** : c'est le cas le plus général de la catégorisation à n classes. Le système doit affecter 0, 1 ou plusieurs catégories à un même document. Ce type de catégorisation correspond par exemple au problème d'affectation automatique des codes CIM aux comptes rendus médicaux.

- **Catégorisation multi catégories disjointes** : c'est une catégorisation à n classes mais le document doit être affecté à une et une seule catégorie. On trouve ce type de catégorisation, par exemple, dans le routage de courriels.

1.4- Processus de catégorisation :

D'après Sahami et al. (1998) la construction d'un système de catégorisation, repose sur trois principales étapes : la représentation de textes, 1 apprentissage par l'enchaînement d'un algorithme de catégorisation (élaboration d'un modèle de prédiction) et enfin, l'évaluation en fonction du modèle généré (Sahami, et al., 1998).

Pour commencer la mise en place d'un processus de catégorisation de textes, il est important de disposer d'un corpus, préalablement étiqueté manuellement. Ce corpus sera divisé en deux catégories: un corpus d'apprentissage, et un autre de test. On peut distinguer trois phases dans le processus de catégorisation de textes (voir Figure 1.2) : l'indexation des documents la construction du classifier et enfin l'évaluation du classifier.

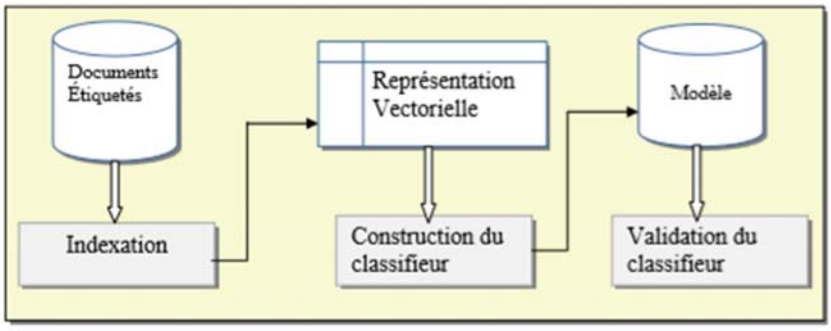

Figure 1.2-Processus de catégorisation de textes.

L'indexation des documents (textes), qui a pour objectif la production d'une représentation exploitable par les algorithmes d'apprentissage, comprend le choix des termes pour la représentation des textes (mot, racine, lemme, ...), la pondération et la réduction des termes.

- ✓ La représentation traduit les documents dans un format spécifique, car il doit être à la fois représentatif de son contenu et amputable par les algorithmes de classification.
- ✓ La pondération consiste à calculer pour chaque terme retenu son poids dans chaque document.

- ✓ La réduction consiste à extraire que les termes qui sont pertinents pour la catégorisation. Cette sélection de caractéristiques est censée aussi réduire la complexité des algorithmes de classification.

La seconde étape du processus de catégorisation consiste à adapter un algorithme de classification au problème abordé. Depuis les années 1990, le problème est abordé avec L'application d'apprentissage automatique. Dans cette étape, on vise à construire des modèles ou classifies qui vont apprendre par eux-mêmes à prédire la classe des documents.

Après cette phase, le classifier peut procéder à la classification. Et enfin, on applique les mesures de performances pour évaluer la qualité de prédiction du classifier.

1.5- Représentation de textes :

La représentation de textes est la phase la plus importante dans le processus de catégorisation de textes par ce que les algorithmes d'apprentissage ne sont pas capables de traiter directement les textes, plus précisément les données non-structurées comme les images, les sons, les vidéos. C'est pour cette raison, qu'on doit opter pour une façon efficace de représenter les instances à traiter (les textes).

La représentation la plus simple d'un texte est introduit dans le modèle vectoriel, qui porte le nom de « Sac de mots » ou « bag of words » (Salton, et al., 1975). L'idée est de transformer le ou les textes en vecteurs où chaque composante représente le poids d'un terme

On transforme un texte d_j= {t1, t2,….,tp} en un vecteur di = (Wi1,Wi2,……,Wip) – Le poids Wij correspond à la contribution du terme tj à la description du texte dj. Il faut signaler que la représentation par vecteur peut entraîner une perte d'information, notamment celle relative à la position des mots dans la phrase.

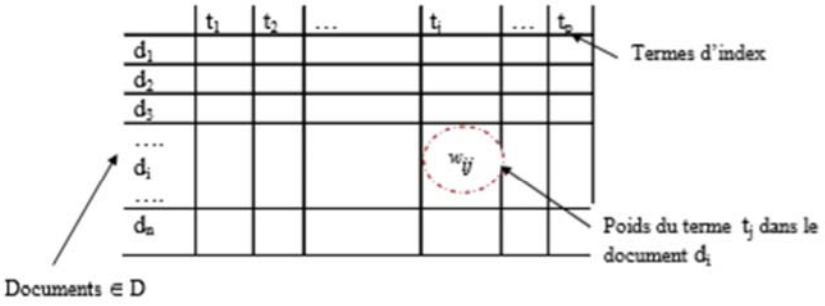

Figure 1.3 Représentation vectorielle des données textuelles.

Come illustré dans la Figure 1.3, F ensemble des textes est transformé en un ensemble de vecteurs, ou un tableau croisé (individus variables) où les individus sont les lignes du tableau et ils représentent les documents, et les variables sont les colonnes de ce tableau et représentent les termes (ou mot d'index) qui sont extraits des documents d'apprentissage pendant la phase d'indexation. Chaque cellule dans le tableau contient le poids du terme dans un document donné (§.2.4).

Pour représenter les documents textuels, plusieurs méthodes sont utilisées, ci-après nous présentons quelques-unes :

- Sac de mots
- Groupe de mots
- Racine ou lemme
- Ngramme de caractère

1.6- Représentation des données textuelles :

1.6.1- Sac de mots :

Dans cette représentation, les termes sont les mots qui constituent un texte. Dans les langues comme le français ou l'anglais, les mots sont séparés par des espaces ou des signes de ponctuations; ces derniers, tout comme les chiffres, sont supprimés de la représentation. Les composantes des vecteurs peuvent être une fonction de l'occurrence des mots dans le texte. Cette représentation exclue toute analyse grammaticale et toute notion de distance entre les mots, et c'est pourquoi elle est appelée « sac de mots » (Harish, et al.. 2010). Avec cette approche, les documents sont représentés par des vecteurs de dimension égale à la taille du vocabulaire, qui est en général assez grande. En effet, même des collections de documents de taille moyenne peuvent contenir de nombreux mots différents, et des vocabulaires de plusieurs dizaines de milliers de mots sont désormais communs. Or la grande dimension de ces données rend la plupart des algorithmes de classification difficiles à utiliser.

1.6.2- Groupe de mots :

Certains auteurs proposent d'utiliser les groupes de mots comme unité de représentation (Fuhr, et al., 1991), (Tzeras, et al., 1993). Les groupes de mots sont plus informatifs que les mots simples, car ils ont l'avantage de conserver l'information relative à la position du mot dans le groupe de mots (Johannes, et

al., 1998), (Fernanda, et al., 2000). Par exemple « recherche d'information », «world wide web», ont un degré plus petit d'ambiguïté que les mots constitutifs.

Normalement, une telle représentation doit décrocher des résultats plus performants que la précédente (sac de mot), mais Lewis (Lewis, 1991), (Lewis, 1992) a constaté que cette représentation n'a pas pu améliorer son système de catégorisation. Il signale que beaucoup d'expériences ne sont pas convaincantes car, il explique que, si les qualités sémantiques sont conservées, les qualités statistiques sont largement dégradées et le grand nombre de combinaisons possibles entraîne des fréquences faibles et trop aléatoires.

1.6.3- Racine ou lemme :

Dans le modèle de la représentation en sac de mots, chaque flexion du mot est considérée comme un terme différent et donc une dimension de plus, ainsi que les différentes formes d'un verbe qui constitue autant de mots. Par exemple : enseigner, enseignement, enseignant, enseignée, enseignés, enseignera, etc. Ces mots sont considérés comme des termes différents, alors qu'il s'agit de la même racine enseigne. Pour la recherche des racines lexicales, il existe plusieurs algorithmes, un des plus connus pour la langue anglaise est l'algorithme de Porter (Porter, 1980).

La lemmatisation consiste à utiliser l'analyse grammaticale afin de remplacer les verbes par leur forme infinitive et les noms par leur forme singulière.

La lemmatisation est donc plus compliquée à mettre en œuvre que la recherche de racines, puisqu'elle nécessite une analyse grammaticale des textes. Un algorithme efficace, nommé Tree Tagger (Schmid, 1994) a été développé pour les langues anglaise, française, allemande et italienne. Cet algorithme utilise les arbres de décision pour effectuer l'analyse grammaticale, avec des fichiers de paramètres spécifiques à chaque langue.

1.6.4- N-grammes de caractères :

Un n-gramme est une séquence de n caractères, c'est donc une chaîne de n caractères consécutifs. La notion de n-grammes a été introduite par Shannon en 1948; il s'intéressait à la prédiction d'apparition de certains caractères en fonction des autres caractères (Shannon, 1948). Depuis cette date, les n-grammes sont utilisés dans plusieurs domaines comme l'identification de la parole, la recherche documentaire, l'identification de la langue etc. (Jalam, et al., 2001). Dans les

recherches récentes, elle est utilisée pour l'acquisition et l'extraction des connaissances dans les corpus. De nombreux travaux (Rahmoun, et al., 2007), (Fümkranz, 1998) utilisent les n-grammes de caractères comme méthode de représentation de documents d'un corpus pour la catégorisation de textes.

De nombreux travaux ont utilisé les n-grammes comme descripteurs de documents pour leur classification. Les auteurs Junker et al. (1997) présentent une étude sur une représentation fondée sur les n-grammes de caractères pour évaluer la classification de textes issus d'OCR et les textes non-OCR (Junker, et al., 1997). Dans (Jalam, et al., 2001) nous trouvons les raisons pour lesquelles les n-grammes donnent des résultats intéressants. Par exemple :

- Les n-grammes permettent de capturer automatiquement la racine des mots les plus fréquents. Il n'est pas nécessaire d'appliquer une étape de recherche de racine et ou de lemmatisation.
- Ces descripteurs sont indépendants de la langue employée dans le corpus. Il n'est pas nécessaire d'utiliser des dictionnaires, ni de segmenter les documents en mots.
- Les n-grammes sont tolérants aux fautes d'orthographe et aux déformations causées lors de la reconnaissance de documents. Lorsqu'un document est reconnu à 1 aide du système OCR il y a souvent une part non négligeable de bruit. Par exemple, il est possible que le mot "feuille" soit lu "teuille". Un système fondé sur les n-grammes prendra en compte les autres n-grammes comme "eui", "uil", etc.

1.7- Pondération des termes :

Une fois que l'on choisit les composantes du vecteur représentant les documents d_j €D, il faut décider de la façon d'associer un poids à chaque coordonnée de leurs vecteurs dj

De nombreuses solutions ont été proposées dans la littérature pour coder les composantes des vecteurs, c'est-à-dire pour attribuer un poids Wij à chaque terme (Salton,1990). Ces méthodes sont basées sur les informations suivantes:

- Plus le terme tj est fréquent dans un document d_i, plus il est en rapport avec le sujet de ce texte.
- Plus le terme tj est fréquent dans la collection, moins il sera utilisé comme discriminant entre textes.

1.8- Pondération booléenne :

Cette pondération est la plus simple représentation des données textuelles. Dans cette pondération le poids Wij vaut 1 si le terme tj apparaît au moins une fois dans le Document dj sinon il vaut 0.

En considérant les trois phrases" comme trois documents :

- {d1) : Le chat mange la souris qui n'a pas eu le temps de manger son fromage.
- (d2) : Le chat n'a plus faim et va rejoindre les autres chats.
- (d3) : Le chien aboie après les chats et les souris mangent le fromage.

Pour chaque document, une représentation vectorielle basée sur les lemmes sera utilisée avec élimination de mots vides.

Le Tableau 1 montre une représentation matricielle croisant en ligne les documents et en colonnes, les lemmes.

	aboyer	*Chat*	*chien*	*faim*	*Fromage*	*manger*	*rejoindre*	*souris*	*temps*
$\mathbf{d_1}$	0	1	0	0	1	1	0	1	1
$\mathbf{d_2}$	0	1	0	1	0	0	1	0	0
$\mathbf{d_3}$	1	1	1	0	1	1	0	1	0

Tableau 1 -Exemple d'une représentation vectorielle booléenne

1.9- Pondération fréquentielle

Elle prend en compte le nombre d'occurrences d'une tenue dans un texte. Cette mesure repose sur l'idée que plus une tenue apparaît dans un texte, plus il est important. En reprenant l'exemple précédent, nous obtenons la représentation du Tableau 1.2 ci-dessous :

	aboyer	*Chat*	*chien*	*faim*	*Fromage*	*manger*	*rejoindre*	*souris*	*temps*
$\mathbf{d_1}$	0	1	0	0	1	2	0	1	1
$\mathbf{d_2}$	0	2	0	1	0	0	1	0	0
$\mathbf{d_3}$	1	1	1	0	1	1	0	1	0

Tableau 1.2 Exemple d'une représentation vectorielle fréquentielle

Une telle présentation est généralement normalisée afin d'éviter de défavoriser les documents les plus longs, contenant ainsi plus de termes. La fréquence du terme tj dans le document dj peut être calculée par la formule suivante:

$$TF(t_j, d_i) = \frac{\#(t_j, d_i)}{\sum_{k=1,p} \#(t_k, d_i)}$$

1.10- Pondération TFIDF:

Elle a été introduite dans le cadre du modèle vectoriel, elle donne beaucoup d'importance aux mots qui apparaissent souvent à l'intérieur du même texte, ce qui correspond bien à l'idée intuitive que ces mots sont plus représentatifs. Mais sa particularité est qu'elle donne également moins de poids aux mots qui appartiennent à plusieurs textes: pour refléter le fait que ces mots ont un faible pouvoir de discrimination entre les classes. Cette pondération issue du domaine de la recherche d'informations tire son inspiration de la loi de Zipf introduisant le fait que les ternies les plus informatifs d'un corpus ne sont pas ceux apparaissant le plus dans ce corpus. Ces mots sont la plupart du temps des mots outils. Par ailleurs, les mots les moins fréquents du corpus ne sont également pas les plus porteurs d'informations (Béchet, 2009). Le poids d'un terme tj dans un document d_j est calculé Comme suit :

$$TFIDF(t_j, d_i) = TF(t_j, d_i) \times \log \frac{N}{DF(t_j)}$$

	aboyer	*Chat*	*chien*	*Faim*	*Fromage*	*manger*	*rejoindre*	*Souris*	*Temps*
$\mathbf{d_1}$	0	0	0	0	0,18	0,35	0	0,18	0,48
$\mathbf{d_2}$	0	0	0	0,48	0	0	0,48	0	0
$\mathbf{d_3}$	0,48	0	0,48	0	0,18	0,18	0	0,18	0

Tableau 1.3 Exemple d'une représentation TFIDF

La fonction TFIDF a démontré une bonne efficacité dans des tâches de catégorisation de textes, et en plus, son calcul est simple (Sebastiani, 2006).

Le codage TFIDF ne corrige pas la longueur des documents. Pour ce faire, le TFIDF est normalisé. On corrige les longueurs des textes par la normalisation en cosinus, pour ne pas favoriser les documents les plus longs.

1.11- Réduction des dimensions

Un problème principal pour l'approche statistique de la catégorisation de textes est la grande dimension de l'espace de représentation. Dans le cas de représentation en sac de mots, chacun des mots d'un corpus est une tenue potentielle, or pour un corpus de taille raisonnable, ce nombre peut être de plusieurs dizaines de milliers. Pour beaucoup d'algorithmes d'apprentissage, il s'agit de sélectionner un sous ensemble de ces tenues, sinon deux problèmes se posent :

- **Le coût du traitement :** le nombre de termes intervient dans l'expression de la complexité de l'algorithme, plus ce nombre est élevé, plus le temps de calcul est important.
- **La faible fréquence de certains termes :** on ne peut pas construire des règles fiables à partir de quelques occurrences dans l'ensemble d'apprentissage.

Les méthodes utilisées pour la réduction de dimension sont issues de la théorie de l'information et de l'algèbre linéaire. Sebastiani (2002) Classe ces techniques de la façon suivante :

Réduction locale: Il s'agit de proposer pour chaque catégorie c_k, telle que c_k €{c,c_2,…, c_m} un nouvel ensemble de tenues T_k <zT. Ainsi chaque catégorie c_k possède son propre ensemble de tenues et chaque texte di sera présenté par un ensemble de vecteurs {d_i} différents selon la catégorie. Pour plus de détail, on peut se référer à (Lewis, et al.. 1994).

1.12- Réduction globale:

Dans ce cas, le nouvel ensemble de tenues T est choisi en fonction de toutes les catégories. Et chaque texte do sera représenté par un seul vecteur quel que soit la catégorie (Yang, et al., 1997), (Caropreso, et al., 2001).

La réduction pennet d'éliminer les informations non-pertinentes et redondantes selon le critère utilisé. Ces principaux objectifs sont :

- Faciliter la visualisation et la compréhension des doimées.
- Réduire l'espace de stockage nécessaire.
- Réduire le temps d'apprentissage et d'utilisation.
- Identifier les facteurs pertinents.

1.13- Sélection des termes :

L'intérêt de la sélection de ternies est triple. D'une part, elle pennet d'écarter les tenues non pertinents d'un point de vue statistique. D'autre part, elle pennet d'éviter le sur apprentissage (Saldarriaga. 2010). Enfin, elle permet d'améliorer l'efficacité des algorithmes d'apprentissage ayant des difficultés à gérer un espace de représentation important.

1.14- Conclusion :

Dans ce chapitre nous avons défini le concept de la catégorisation et la représentation du texte et nous avons introduit les notions nécessaires à la compréhension de chapitre suivant : classification de texte et l'apprentissage automatique superviser (chapitre 2).

Chapitre 2

2- Techniques de classification (Techniques d'apprentissage automatique) :

La notion d'apprentissage automatique fait référence au développement, à l'analyse et à l'implémentation des méthodes qui permettent à une machine d'évoluer grâce à un processus d'apprentissage et de remplir des tâches dont il est difficile ou impossible de remplir par des moyens algorithmiques classiques.
L'apprentissage automatique englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle.
Nous distinguons différents types d'apprentissage : apprentissage non supervisé et apprentissage supervisé.

2.1- Apprentissage supervisé :

Il relève d'une démarche inductive consistant à construire automatiquement un classifier qui apprend, à partir des exemples déjà classés (ou étiquetés), les caractéristiques et les propriétés des catégories cibles. Ce type d'apprentissage est dit supervisé par ce que la fonction de classification s'entraine sur les catégories (ou classe) ainsi que sur leurs caractéristiques.
La classification supervisée cherche à prédire l'appartenance de documents à des classes connues à priori. Ainsi, c'est l'ensemble des techniques qui visent à deviner l'appartenance d'un individu à une classe
Dans ce mémoire nous nous intéressons qu'à l'apprentissage supervisé dans le cas de la catégorisation de textes qui consiste à apprendre à partir d'un ensemble d'exemples une fonction de prédiction. Cette fonction permettra par la suite de prédire la classe (ou la catégorie) de chaque nouveau cas (ici le texte). La Figure (2) présente le principe de l'apprentissage supervisé dans le cas de filtrage de spam :

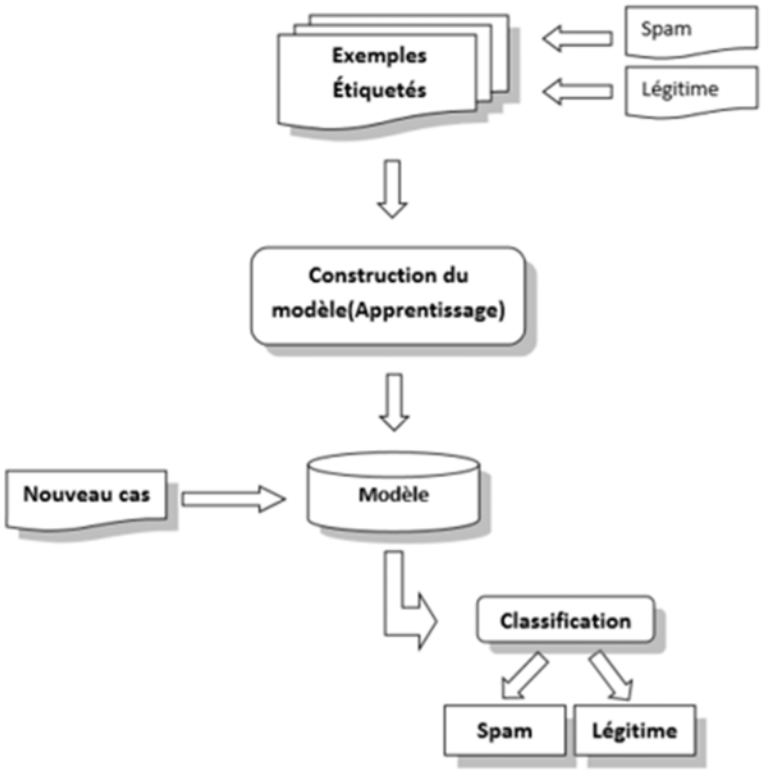

Figure 2 : Filtrage de spam à base d'apprentissage supervisé.

2.2- Algorithmes d'apprentissage supervisé :

Dans le courant de l'apprentissage supervisé, différents types de classifieurs ont été mis au point, dans le but d'atteindre un degré maximal de précision et d'efficacité, chacun ayant ses avantages et ses inconvénients. Parmi les algorithmes d'apprentissage supervisé existants, on peut citer :

2.2.1- L'algorithme Naïve Bayes :

Le modèle probabiliste Naïf de Bayes (NB) qui est le représentant le plus populaire des classifieurs probabilistes, est fondé sur le théorème de Bayes.
Ce modèle vise à estimer la probabilité conditionnelle d'une catégorie sachant un document et affecte au document là (ou les) catégorie(s) la (les) plus probable(s). La partie naïve de ce modèle est l'hypothèse d'indépendance des mots, c'est-à-dire que la probabilité conditionnelle d'un mot sachant une catégorie est supposée indépendante de cette probabilité pour les autres mots
Considérons $\vec{di}$= (wi1,wi2 ,……. ,wip) la représentation vectorielle représentant un texte di et C = {c1,c2 ,….,cm} un ensemble de classes. En s'appuyant sur le

théorème de Bayes, la probabilité que ce document appartienne à la classe c k dans notre cas C k ={spam,ham} est définie par :

$$P(C_k / di) = \frac{P(C_k) \times P(di / C_k)}{P(di)}$$

Le but étant de discriminer les différentes classes, il suffit donc d'ordonner P (C_k/d_i) pour toutes les classes. On peut alors supprimer le dénominateur P (d_i) qui est le même pour toutes les classes. P (C_k) est la probabilité à priori qui est estimée par le pourcentage d'exemples appartenant à la classe C_k dans le corpus d'apprentissage.

En faisant l'hypothèse que les termes sont indépendants, la probabilité conditionnelle P (d_i/c_k) est définie par :

$$P(di/C_k) = \prod_{j=1,p} P(Wij/C_k)$$

La classe c k d'appartenance de la représentation vectorielle $\vec{di}$ d'un document di est définie par :

$$C_{NB} = \arg\max P(C_k) \prod_j P(Wij/C_k)$$

2.2.2- Les K voisins les plus proches :

Cette méthode a prouvé son efficacité face au traitement des données textuelles. La phase d'apprentissage consiste à stocker les exemples étiquetés. Le classement de nouveaux textes s'opère en calculant la similarité6 entre la représentation vectorielle du document et celle de chaque exemple du corpus d'apprentissage. Les k éléments les plus proches sont sélectionnés et le document est assigné à la classe majoritaire (le poids de chaque exemple dans le vote étant éventuellement pondéré par sa distance).

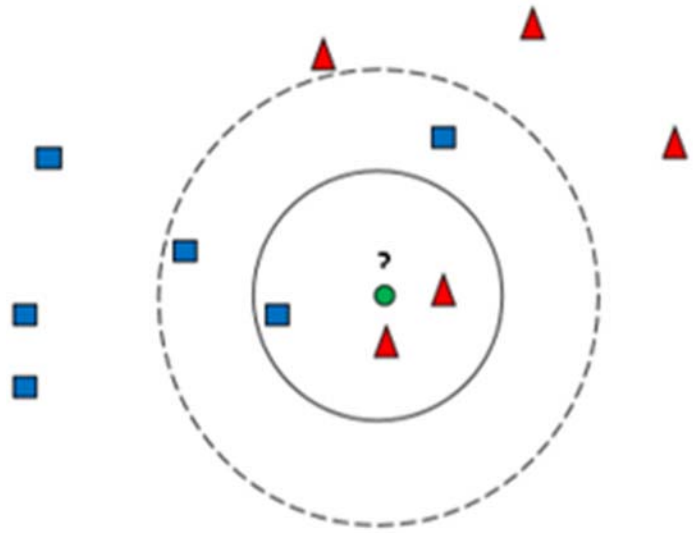

Figure 2.1 : K voisins les plus proches

Le choix de la valeur de k est dépendant de la taille de l'échantillon et des classes, et influence les résultats de la classification. Dans l'exemple de la Figure 1.5, l'objet rond sera classifié triangle si k=3 et classifié Carré si k=5.

Lorsque « k » est petit, la classification est plus sensible à cause des documents appartenant à une classe mais dont leur vecteur de représentation ressemble beaucoup plus à une autre. Par contre, lorsque « k » est trop grand, les catégories ayant peu d'exemples peuvent être désavantagées par rapport à celles qui en ont plus. On peut remédier à cela en pondérant le vote par la distance qui sépare les plus proches voisins de l'individu à classer.

Si la qualité de catégorisation obtenue par les k plus proches voisins (K-ppv) est satisfaisante que celle obtenue avec d'autres méthodes qui nécessitent un apprentissage complexe, le temps nécessaire à son déroulement peut être un obstacle difficilement incontournable ; là où la complexité des autres méthodes est fonction du nombre de catégories.

2.2.3- Les machines à support vectoriels (SVM) :

Le but de SVM est de trouver un classificateur qui sépare au mieux les données et maximise la distance entre ces deux classes. Ce dernier est un classificateur linéaire appelé hyperplan. Comme montré dans la Figure(2.2), cet hyperplan sépare les deux ensembles de points.

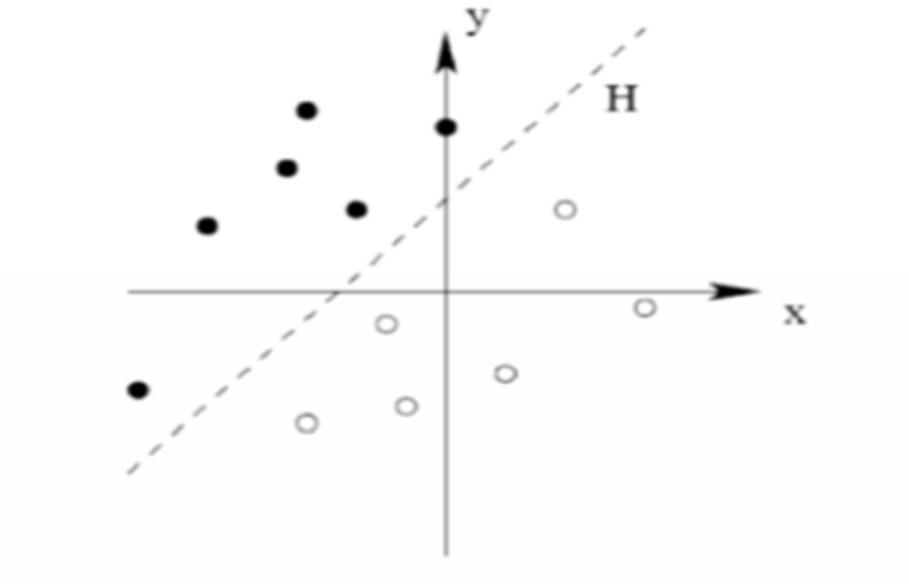

Figure 2.2 : machines à support vectoriels.

Les points les plus proches, qui seuls sont utilisés pour la détermination d'hyperplan, sont appelés vecteurs de support Voir la figure (2.3).

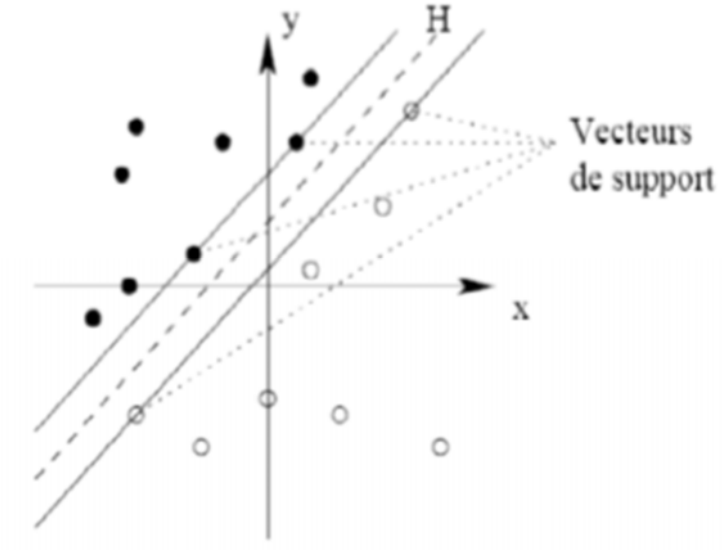

Figure 2.3 : machines à support vectoriels 2.

2.3- Remarques sur les algorithmes d'apprentissage supervisé (NB ; KNN ; SVM) :

1. **Le classifier naïf de Bayes** s'est montré très performant pour des tâches de classification de Textes comme le filtrage de spam que d'autres méthodes malgré son hypothèse d'indépendance des mots. Il reste capable de bien fonctionner avec des données incomplètes comme il peut être appliqué à de nombreux secteurs d'activité: médicale, juridique, etc
2. **Les SVM** donnent de très bons résultats de classification de textes mais sont très coûteux en temps d'apprentissage et possèdent une limitation théorique, Le modèle sous-jacent à se classifier a été conçu pour la classification binaire : il cherche un plan séparateur qui sépare l'ensemble des objets en deux classes.
3. **Les K-ppv** sont très simples à mettre en œuvre, et permettent une implémentation rapide pour fournir des résultats satisfaisants. Cette méthode reste robuste sur des cas de données incomplètes, mais elle est très couteuse en temps de classification et stockage mémoire.

2.4- Conclusion :

Dans ce chapitre, nous avons exposé la tache de classification des textes avec leur technique, à travers la représentation des textes et la catégorisation de chapitre (1). Finalement nous avons présenté quelques algorithmes d'apprentissage automatique supervisé et en donnant quelque remarques avec des principes et inconvénients de chaque algorithme d'apprentissage présenter.

Chapitre 3

3- Introduction :

Le spam est un grand problème pour les internautes. Les augmentations récentes du taux de spam ont causé une grande inquiétude parmi la communauté Internet. De nombreuses solutions avaient été suggérées pour résoudre le problème.

Dans ce chapitre, nous présentons tout d'abord les débuts du spam, ses objectifs, ses contenus, ses impacts et les différentes techniques utilisées pour détecter ce type de courriels.

3.1- Naissance et débuts du spam :

3.1.1- Origine du mot spam(Historique) :

En 1937 La société Hormel Foods1 organise un concours pour trouver un nouveau nom pour leur jambon épicé, Ce nom doit être aussi caractéristique que le goût du produit « Spiced Ham » et qui propose « Spam » pour ce produit, fut donc la marque retenue.

Cette viande précuite en boîte souvent synonyme de mauvaise nourriture a été largement utilisée par l'intendance des forces armées américaines pour la nourriture des soldats pendant la Seconde Guerre mondiale et sera introduite dans diverses régions du monde à cette occasion.

3.1.2- Evolution du spam :

C'est dans les années 90 que les personnes malveillantes ont eu l'idée de rentabiliser l'envoi massif de pourriels. Au fur et à mesure que le temps passe les techniques utilisées par les spammeurs sont évolué avec les filtres anti spam mis en service sur le marché.

Autourde1994, les spams plus notables ont commencé à se réaliser. Les spams les plus notables dans l'histoire, ou les plus parlés, étaient le Spam de Canter et Siegel et le spam de Michael Wolff and Company Inc (a décidé de commencer à spammer pour promouvoir certains des livres de Wolff).

3.2- Définition du spam :

Le spam est un message électronique non sollicité, envoyé massivement à un grand nombre de destinataires, à des fins publicitaires ou malveillantes.

Le terme spam est aussi utilisé pour désigner le même type de message transmis par d'autres moyens de communication électroniques tels que les messageries instantanées, les blogs, les forums, et plus récemment, des réseaux de téléphonie mobile, via les SMS ou MMS. Même si le moyen de communication est différent, les techniques d'envoi et de détection restent relativement similaires.

Le premier spam (Figure 3) date du 3 mai 1978. Ce jour-là, sur le réseau ARPANET2, Gary Thuerk, commercial de la société informatique DEC3, invitait par e-mail 393 personnes à découvrir sa nouvelle machine, le 2020.

Mail-from: DEC-MARLBORO rcvd at 3-May-78 0955-PDT
Date: 1 May 1978 1233-EDT
From: THUERK at DEC-MARLBORO
Subject: ADRIAN@SRI-KL

WE INVITE YOU TO COME SEE THE 2020 AND HEAR ABOUT THE DECSYSTEM-20 FAMILY AT THE TWO PRODUCT PRESENTATIONS WE WILL BE GIVING IN CALIFORNIA THIS MONTH. THE LOCATIONS WILL BE:

TUESDAY, MAY 9, 1978 – 2 PM
HYATT HOUSE (NEAR THE L.A. AIRPORT)
LOS ANGELES, CA

THURSDAY, MAY 11, 1978 – 2 PM
DUNFEY'S ROYAL COACH
SAN MATEO, CA
(4 MILES SOUTH OF S.F. AIRPORT AT BAYSHORE, RT 101 AND RT 92)

A 2020 WILL BE THERE FOR YOU TO VIEW. ALSO TERMINALS ON-LINE TO OTHER DECSYSTEM-20 SYSTEMS THROUGH THE ARPANET. IF YOU ARE UNABLE TO ATTEND, PLEASE FEEL FREE TO CONTACT THE NEAREST DEC OFFICE FOR MORE INFORMATION ABOUT THE EXCITING DECSYSTEM-20 FAMILY.

Figure 3 : Premier spam sur le réseau ARPANET2, Gary Thuerk.

3.3- Objectifs et statistiques sur les spam :

Au départ, le spam visait principalement des objectifs publicitaires. Aujourd'hui, il s'est considérablement développé, diversifié et complexifié, pour atteindre de plus en plus souvent des objectifs malveillants. En effet, Le spam s'est non seulement développé en termes de volume, mais également en termes de contenu (voir figure 1.2). Aujourd'hui, les objectifs des spam sont très variés en voici une liste non exhaustive :

- **Hameçonnage (ou phishing)** : L'objectif est de réussir à se faire passer pour un organisme connu par l'utilisateur, dans le but de lui voler des informations à caractère confidentiel. Par exemple, on reçoit un mail provenant "apparemment" de notre banque, ou d'un autre site où l'on dispose d'informations personnelles. Dans ce mail, il est demandé de cliquer sur un lien (pour des motifs divers : Réactualisation, etc.), après avoir cliqué sur ce lien, une page web s'affiche... sur laquelle il est demandé de rentrer ses coordonnées bancaires ou toute autre information personnelle. Parmi les sites Top les plus contrefaits pour les attaques de phishing, on retrouve eBay, Paypal et Bank of America.
- **Publicité** : L'objectif est de vanter les mérites d'un produit quelconque. Il s'agit par exemple de produits pharmaceutiques, de produits de luxe, de logiciels divers et variés, de jeux d'argent. Ils peuvent également soutenir-agate idées politiques, culturelles ou religieuses et / ou organisations.
- **Scam** : Il s'agit d'une attaque basée sur la naïveté des destinataires dans le but de leur soutirer de l'argent. L'exemple le plus courant est le scan nigérien: un dignitaire d'un pays d'Afrique vous demande de servir d'intermédiaire pour une transaction financière importante, en vous promettant un bon pourcentage de la somme. Pour amorcer la transaction, il vous faut donner de l'argent.
- **Canular** : L'objectif est de faire circuler une information semblant très sensible, souvent avec un caractère d'urgence : fausse alerte de virus, fausse alerte de contamination potentielle, chaine de solidarité…. Par exemple : « un nouveau virus très dangereux se propage, il faut faire circuler l'information »; « des sous-vêtements sont infectés par une dangereuse bactérie ».
- **Malware** : Est un logiciel conçu pour infiltrer ou endommager un système informatique. Il est communément pris pour contenir des virus

informatiques, vers, chevaux de Troie, spywares et adwares. Ce type de logiciel est souvent envoyé en tant que non suspect d'une pièce jointe. Lorsque l'utilisateur ouvre le fichier, le logiciel malveillant s'installe. L'interdépendance entre les spams et les logiciels malveillants a évolué Spam logiciels malveillants propagation des e-mails, les logiciels malveillants est utilisé pour infecter un hôte de sorte que l'hôte peut être contrôlé à distance et utilisé pour l'envoi de plus de spams. Ces hôtes infectés sont désignés comme des « ordinateurs zombies ». Beaucoup de gens croient que la plupart des spams sont envoyés par des botnets, qui constituent un réseau de PC zombies.

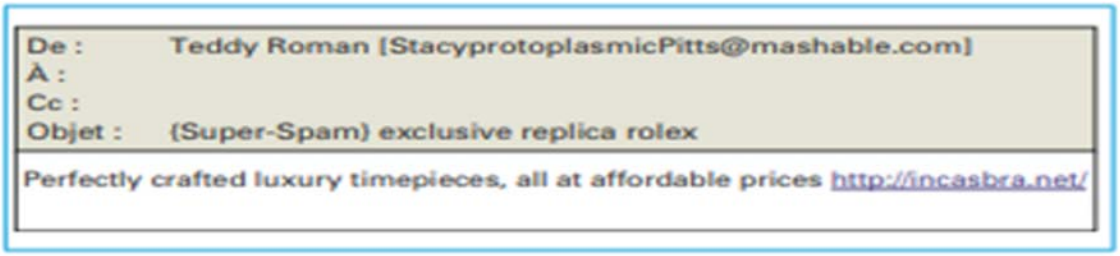

Figure 3.1 : Exemple de spam publicitaire

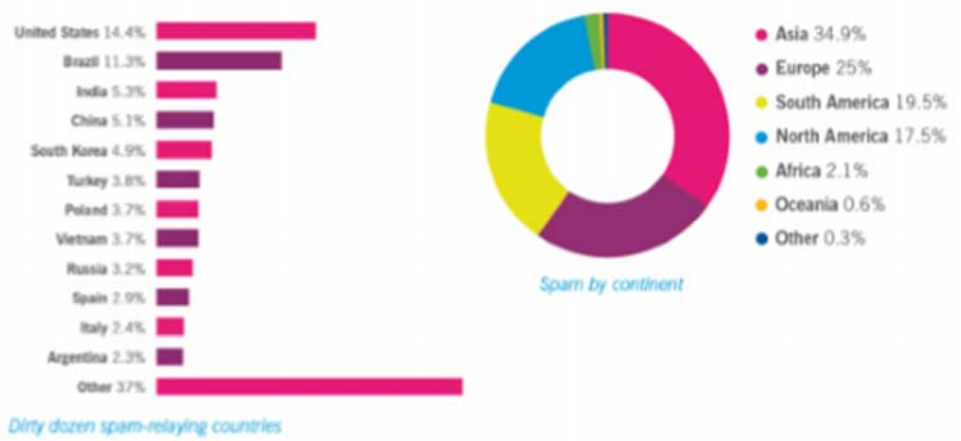

Figure 3.2 –Répartition des spam par contenu & les pays ayant des transmissions avec spam.

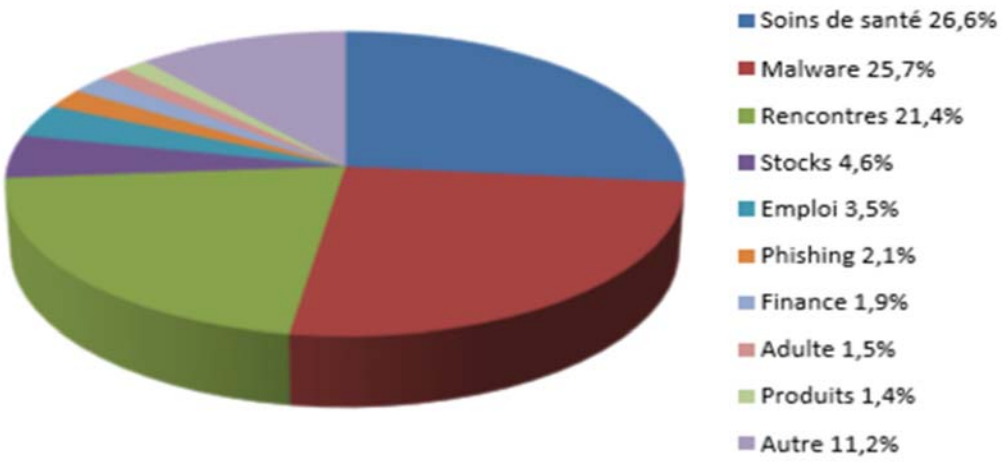

Figure 3.3 Répartition des spam par contenu

On a quelques statistiques sur le taux global de spam entre les années 2012 et 2017. Dans la dernière période il a été constaté que le spam représentait 55% de tous les messages électroniques, comme au cours de l'année précédente.

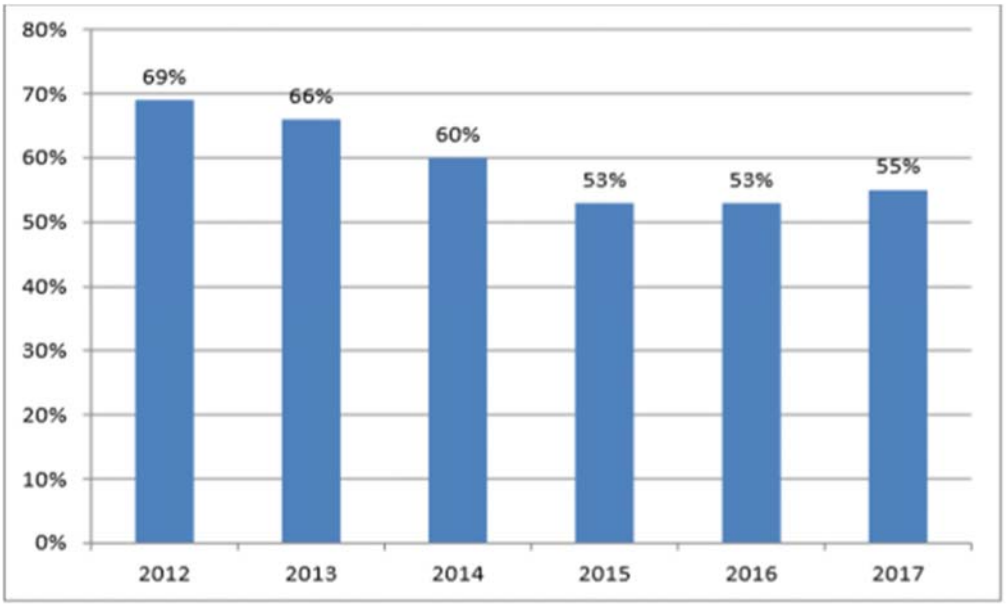

Figure 3.4 : statistiques sur le taux global de spam entre les années 2012 et 2017.

3.4- Impactes du spam sur les utilisateurs et les fournisseurs :

Dans cette section, nous présentons les effets du spam, au niveau des utilisateurs, entreprises et FAI.

3.4.1- Perte de temps :

- Encombrement anormal des boîtes aux lettres.

- Suppression des courriels indésirables.

- Configuration et maintenance des filtres.

- Consultation des courriels rejetés pour y détecter les bons à cause du risque de passer à côté d'emails importants mal catalogués par les outils de détection anti-spam. 1.4.2.

3.4.2- Perte de bande passante et d'espace disque :

- Spécialement pour les utilisateurs de modems.

- Les pièces jointes des virus et spam peuvent être grands. 1.4.3.

3.4.3- Pertes financières non négligeables aux niveaux des entreprises et FAI :

- une augmentation des coûts de gestion opérationnelle et support lié à la gestion anti spam.
- perte de productivité des salariés, Selon une étude, le spam aurait coûté environ 712 $ par employé et par an aux entreprises. À ce chiffre, il faut rajouter 113 à 183 $ par employé et par an pour la gestion des emails en quarantaine.

3.5- Techniques de filtrage du spam :

Plusieurs techniques de lutte contre le spam sont possibles et peuvent être cumulées : analyse statistique (filtre bayésien), filtrage par mots clés, listes blanches, listes noires. Ces techniques de lutte doivent s'adapter en permanence car de nouveaux types de spam réussissent à les contourner.

Deux solutions de détection de spam sont envisageables : la détection au niveau du serveur mail FAI et la détection au niveau de l'utilisateur final.

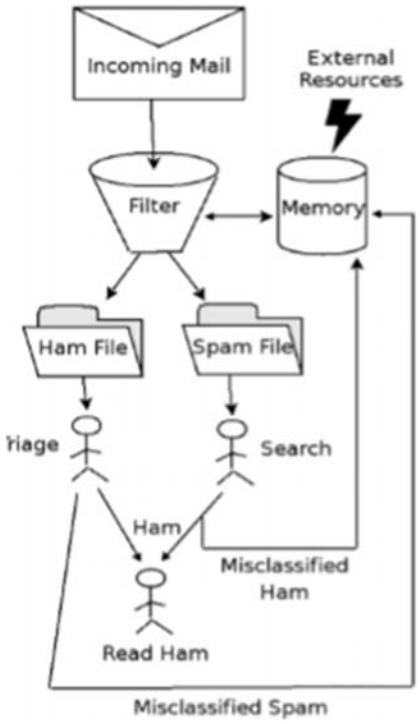

Figure 3.5 utilisation de filtre anti-spam.

Ces outils peuvent être divisés en deux groupes : le filtrage d'enveloppe, et le filtrage de contenu.

3.5.1- Filtrage d'enveloppe :

Ce type de filtrage s'applique uniquement sur l'en-tête du message, qui contient souvent assez d'informations pour pouvoir distinguer un spam. Cette technique appliquée au niveau du serveur FAI présente l'avantage de pouvoir bloquer les courriels avant même que leur corps ne soit envoyé, ce qui diminue grandement le trafic sur la passerelle SMTP.

Dans cette catégorie, nous trouvons les techniques suivantes :

- Filtrage par listes noires.
- Filtrage par listes blanches.
- Filtrage par liste grise.
- Filtrage par vérification du domaine.

3.5.2- Filtrage du contenu :

Ce type de filtrage se fait au niveau de l'utilisateur où son contenu est analysé pour détecter les spam qui ont réussi à passer à travers le filtre d'enveloppe.

Dans cette catégorie nous trouvons les techniques suivantes :

- Filtrage par mots clés.
- Filtrage par caractère.
- Filtrage d'image.
- Filtrage d'URL.
- Filtres bayésiens.
- Machine à Vecteurs de Support (SVM).

3.6- Spamming :

3.6.1- Description :

A nos jours il y a plus 2 milliards d'internautes dans le monde, des milliers de visites sont faites à plusieurs sites Web tous les jours. Chaque année; un nombre très important des messages non sollicités ou des Spams sont expédies dans le monde et de plus en plus sont considérés comme un réel fléau et problème majeur sur Internet.

3.6.2- Les types de spam :

Les Spam publicitaires peuvent être classés en quatre types, qui sont ennuyeux, difficiles, trompeurs et mauvais :

- **Les Spam ennuyeux :** contenant des textes simples, répétés et évidents qui existent déjà dans la base de données de mots clés de spam, ainsi que des liens évidents. Ce type est facilement détecté.
- **Les Spam difficiles :** ce n'est pas très facile à détecter et peuvent contenir des liens cachés, des textes et des hyperliens.
- **Les Spam trompeurs :** Ce type est conçu avec de mauvaises intentions, telles que l'escroquerie ou la fraude, ou lancer des attaques de Phishing. L'exemple d'un spam trompeur un client reçoit un Email contenant le résultat d'un concours sur Facebook et que le client a gagné des cadeaux, et quand le

client clique sur un des cadeaux proposés le navigateur nous dirige vers une page de publicité.

- **Les Spam mauvais :** Le contenu de ce type de spam est conçu pour créer un moyen pour propager des virus, des logiciels malveillants, des vers, des chevaux de Troie et d'autres outils qui soulèvent de sérieuses menaces de sécurité dans la communauté.

3.6.3- Il y a d'autre catégories de spam sont:

Adulte _ Financier _Fraude_ Santé _ Internet_ Loisirs_ 419-spam_ Politique _ Produits.

3.7- Conclusion :

Dans ce chapitre, nous avons présenté quelques définitions sur les spam avec leurs objectifs et impacts et sur le filtrage de spam.et présenté des techniques de filtrage de spam.et on parle de spamming et leurs caractéristiques.

Chapitre 4

4- Bio inspirée et méta heuristique:

4.1- Introduction :

L'informatique inspirée par la biologie (ou l'informatique bio-inspirée) est un domaine d'étude qui regroupe vaguement des sous-domaines liés aux thèmes du connexionnisme, du comportement social et de l'émergence. Il est souvent étroitement lié au domaine de L'intelligence artificielle, dans la mesure où bon nombre de ses objectifs peuvent être liés à l'apprentissage automatique. Il fait largement appel aux domaines de la biologie, de l'informatique et des mathématiques.

L'informatique inspirée biologiquement est un sous ensemble majeur du calcul naturel, L'observation de la nature a conduit les chercheurs à emprunter les principes observés dans une nouvelle ère est ouverte avec les algorithmes inspirés de la nature (bio-inspiré) qui sont des méta-heuristiques imitant la nature pour résoudre des solutions de qualité supérieure.

4.2- Complexité et optimisation :

Théorie de complexité : La théorie de la complexité s'intéresse à l'étude formelle de la difficulté des problèmes en informatique, la question étant de savoir si ces problèmes peuvent être résolus efficacement ou pas en se basant sur une estimation théorique des temps de calcul et des besoins en espace mémoire. On distingue deux grands types de problèmes : les problèmes de décision et les problèmes d'optimisation. [Labed, 2013].

4.3- Problème de décision:

Un problème de décision est un problème qui pose une question dont la réponse est "oui" ou "non" [Labed, 2013].

4.4- Problème d'optimisation:

Un problème d'optimisation cherche à optimiser une certaine valeur [Labed, 2013].

4.5- Les Méthodes de résolution de problèmes :

Il y'a deux grandes classes de méthodes pour la résolution de problèmes :

- Les méthodes exactes,
- Les méthodes approchées.

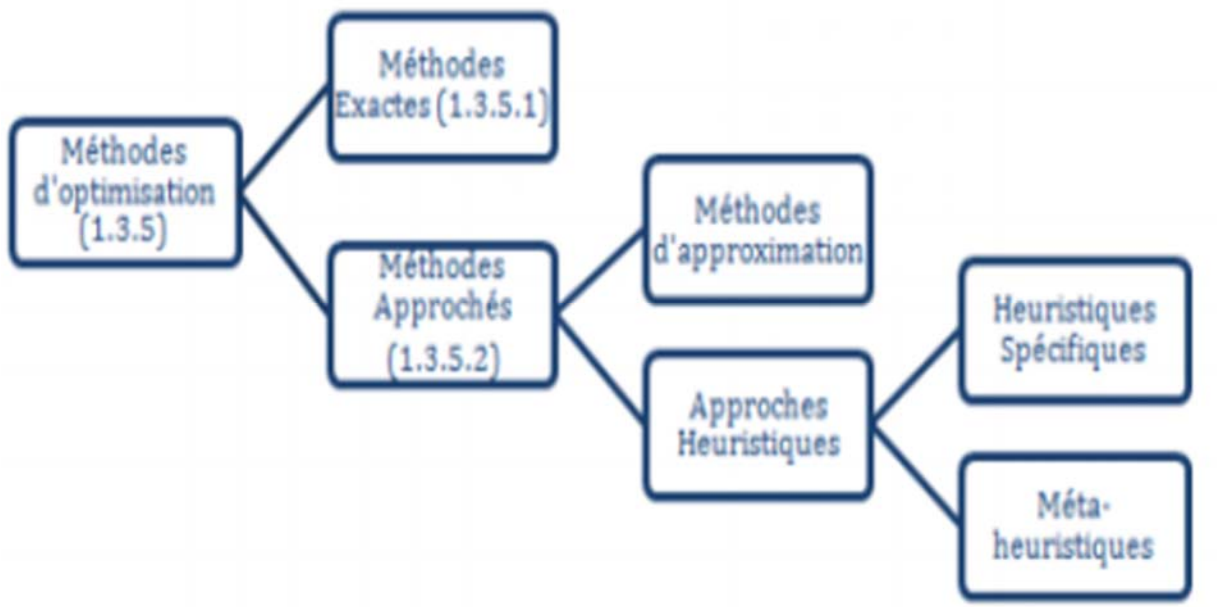

Figure 4 : Classification des méthodes d'optimisation.

4.5.1- Méthodes exactes :

Ce sont des méthodes qui garantissent l'obtention de la solution optimale du problème traité, ils consistant à effectuer une énumération explicite de toutes les solutions pour assurer l'obtention de toutes les solutions ayant le potentiel d'être meilleures que la solution optimale trouvée au cours de la recherche, mais au prix de temps de calcul prohibitif et/ou d'espace mémoire souvent très grand.

4.5.2- Méthodes approchées :

Dans certaines situations, il est nécessaire de disposer d'une solution de bonne qualité en un temps raisonnable, les méthodes approchée souffrants cette possibilité. Ces méthodes peuvent être réparties en deux classes **:**

- **Heuristiques :** Une heuristique est une optimisation alternative méthodes capables de déterminer pas parfaitement solution précise, mais un

ensemble de bonne qualité approximations à solution exacte. Les heuristiques, étaient initialement basées essentiellement sur connaissances et de l'expérience des experts et d'explorer l'espace de recherche dans un contexte particulièrement moyen pratique.

- **Méta-heuristiques Meta** : dans un niveau supérieur, **Heuristique** : trouver Une méta heuristique est formellement définie comme un processus de génération itérative qui guide une heuristique subordonnée en combinant intelligemment des concepts déférents pour explorer et exploiter l'espace de recherche. Des stratégies d'apprentissage sont utilisées pour structurer l'information afin de trouver efficacement des solutions quasi optimales.
- **Les avantages de Meta heuristiques** : Les principaux avantages des métas heuristiques sont résumés par [Hao et al., 1999]:
  - Généralité et application possible à une large classe de problèmes.
  - Simplicité et rapidité.
  - Efficacité pour de nombreux problèmes.
  - Possibilité de compromis entre qualité des solutions et temps de calcul.
- **Classification des Meta heuristiques :**

Il existe plusieurs critères pour classifier les métas heuristiques:

- **Naturel ou non naturel :** Selon ce critère, les métas heuristiques sont divisés en deux classes : le méta heuristique sin spirées des phénomènes naturels et celles qui ne sont pas inspirées.
- **Recherche à base de population ou unique :**

Une autre façon de classifier les métas heuristiques est de distinguer celles qui travaillent avec une population de solutions de celles qui ne manipulent qu'une seule solution à la fois. Les méthodes qui tentent itérativement d'améliorer une solution sont appelées aussi méthodes de recherche locale ou méthodes de trajectoire.

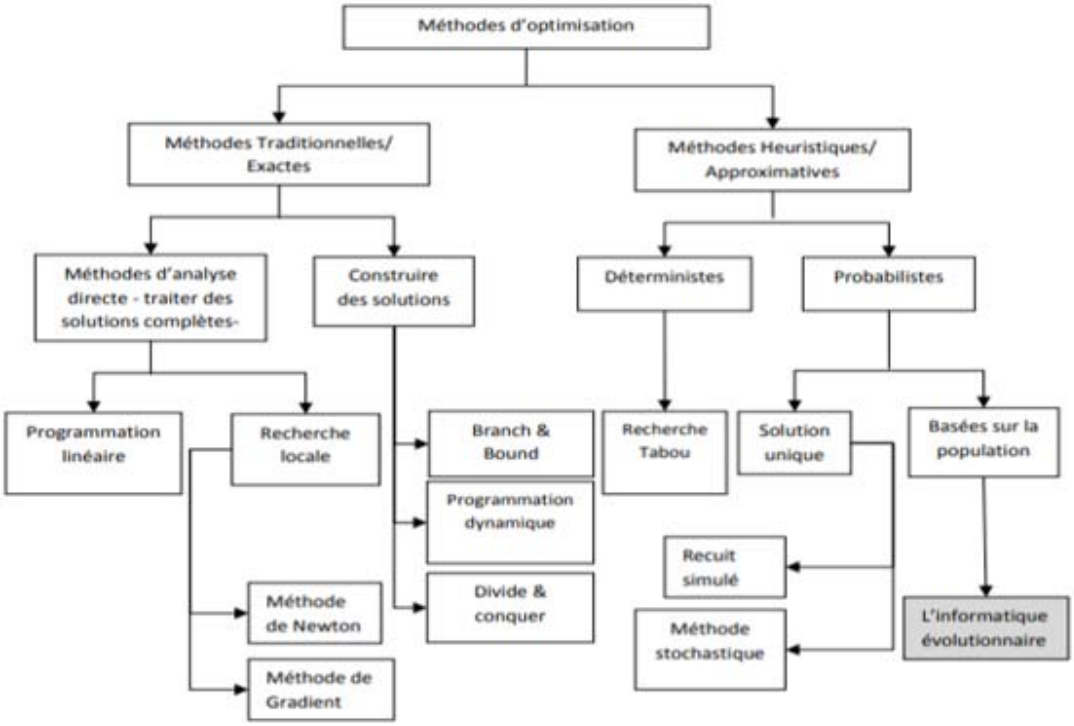

Figure 4.1 Taxonomie des déférentes méthodes d'optimisation.

4.6- Informatique bio-inspirée :

L'informatique inspirée est une méthode de recherche visant à résoudre des problèmes à l'aide de modèles informatiques basés sur les principes de la biologie et du monde naturel. Généralement considérée comme une approche philosophique, l'informatique bio-inspirée est utilisée dans un certain nombre de domaines d'études connexes de l'informatique, plutôt que comme un domaine d'étude lui-même. L'informatique bio-inspirée est une extension du domaine connexe du bio mimétisme.

4.7- Motivation de l'utilisation du bio-inspiré :

- **La réactivité :** les éléments du système coopèrent et communiquent entre eux via des interactions locales. Ils sont capables de réagir instantanément aux changements d'environnement.
- **L'auto-adaptation :** l'aptitude d'un système à modifier ses paramètres de manière que son fonctionnement demeure satisfaisant en dépit des variations de son environnement.

- **L'auto-organisation :** l'organisation interne du système se structure automatiquement sans être dirigée par une source extérieure.
- **La modularité :** le système est composé d'éléments simples qui coopèrent ensemble pour atteindre l'objectif global. Le système est donc évolutif.
- **La décentralisation :** ceci garantit un système robuste, capable de continuer à fonctionner en cas de défaillance d'un de ses composants.
- **Emergence :** les éléments simples qui interagissent vont accomplir des taches extraordinaires. – La simplicité de la mise en œuvre.

4.8- Processus de création d'un algorithme inspiré de la nature :

L'homme s'inspire de la nature pour développer une observation sur un phénomène naturel (Figure :). Il commence par sa modélisation en utilisant des simulations Mathématiques. Une fois le modèle est raffiné, il sera utilisé pour extraire un méta heuristique.

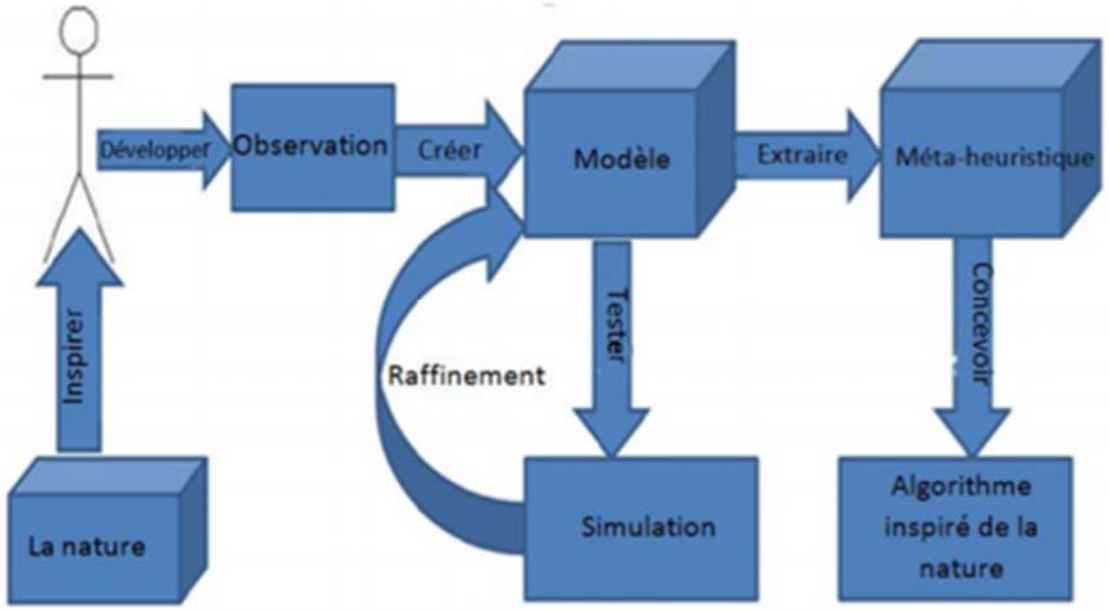

Figure 4.3 : Processus de création d'un algorithme inspiré de la nature.

4.9- Classification d'algorithmes bio-inspirés :

Les méthodes bio-inspirés peuvent être réparties en deux grandes classes selon la source d'inspiration de la méthode bio-inspiré : Algorithmes évolutionnaires et Algorithmes basés essaim :

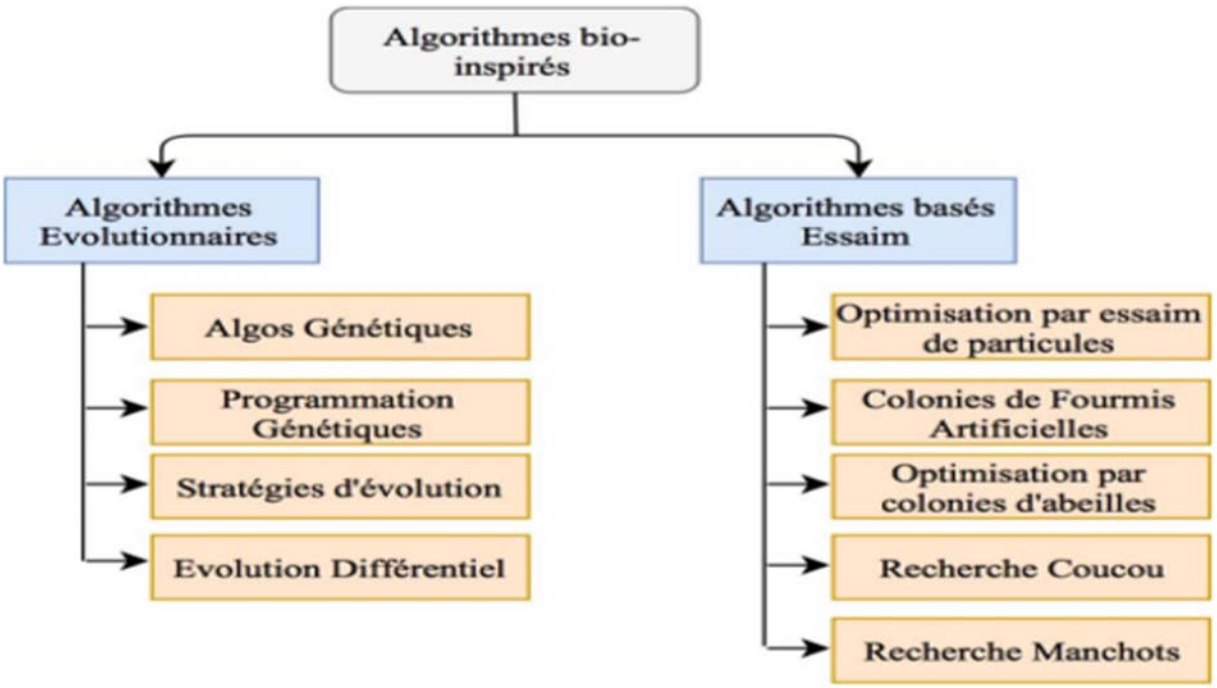

Figure 4.4 –Classification de méthodes bio-inspirées.

4.10- Méthodes bio-inspirées pour la détection de spam :

4.10.1- Optimisation par essaim de particule PSO :

Le filtrage de spam peut être considéré comme un problème de classification de textes qui consiste à attribuer à des documents textuels des classes prédéfinies. Plusieurs techniques de résolution de ce problème ont été proposées. Cependant, les algorithmes d'apprentissage automatiques pour la détection des spams restent toujours en développement. Appliquent l'algorithme d'optimisation par essaim de particule (PSO) pour le filtrage de spam et les résultats obtenus sont satisfaits.

4.10.2- Système immunitaire artificiel :

Le système immunitaire est le mécanisme de défense de l'organisme. Il détecte les microbes, les virus et les substances toxiques afin de les éliminer. Dans une recherche en 2005, Oda a inspiré de ce principe pour résoudre le problème de détection de spam.

4.10.3- Optimisation par colonies de fourmis :

L'algorithme d'optimisation par colonies de fourmis est parmi les premiers algorithmes bio-inspirés. Il a été utilisé pour la résolution de plusieurs problèmes d'optimisation. En 2012, Taweesiriwate utilise cet algorithme pour le problème de détection de spam.

4.11- Conclusion :

Dans ce chapitre nous avons essayé de présenter un état de l'art sur les méthodes bio-inspiré pour la détection de spam commençant par connaître au mieux les concepts liés à la complexité et l'optimisation, ainsi les méthodes de résolution exactes et approchées. Nous avons présenté quelque algorithmes bio-inspirés. Et à la fin, nous avons essayé de capter les différentes méthodes bio-inspirées utilisées pour la détection de spam.

Chapitre 5

5- Contribution :

5.1- Introduction :

Il existe plusieurs métas heuristiques inspirés de la nature qui ont donné des meilleurs résultats dans le domaine de l'intelligence artificielle. Nous avons exploré deux approches bio-inspirée (social worker bees (SWB) et Artificial Social Roaches (ASR) pour la détection du Spam. Ce chapitre est devisé en deux parties : la première présente les méthodes bio inspirée proposés et la deuxième contient les résultats de l'implémentation et en utilisant le corpus smsSpamCollection. Basant sur ces résultats, on le discute et on va sélectionner le meilleur algorithme parmi les algorithmes étudiés pour la classification des emails reçus.

5.2- La méthode bio inspirée proposés :

5.2.1- Algorithm 1: Artificial Social Roaches (ASR):

Les cafards sociaux artificiels (ASR) proposés par Bouarara & al in (Bouarara, 2015) basés sur le mode de vie des cafards et le phénomène social de se cacher sous l'abri avec moins de luminosité là où il y a plus de ses congénères de la même colonie. Semblable à d'autres algorithmes d'optimisation, l'objectif est de trouver la solution globale d'un problème d'optimisation. L'ASR fonctionne comme suit: initialement N cafards sont placés dans un espace de recherche et N abris sont sélectionnés à l'avance. L'attraction de chaque abri pour chaque cafard est calculée en utilisant deux règles de dissimulation (shelter darkness « obscurité de l'abri » et security quality « qualité de sécurité »), le but est que chaque cafard doit se cacher sous l'abri le plus attrayant. Par la suite l'algorithme va évoluer d'un temps T à un temps T + 1 et chaque cafard va passer d'une position à une autre jusqu'à atteindre l'abri où il se sent plus en sécurité. Plus de détails concernant la source d'inspiration et le modèle biologique de l'ASR sont décrits dans (Bouarara, 2015).

Artificial Social Roaches (ASR) pour la détection de spam L'ASR peut fonctionner comme un filtre anti-spam en suivant un ensemble d'étapes comme le présente la figure 5. Comme le montre la figure ci-dessous, les cafards artificiels sont les messages. Les différentes étapes d'application de l'algorithme ASR pour le filtrage anti-spam sont détaillées dans ce qui suit :

i. **Extraction de termes (TE) et vectorisation de messages (MV)** : Les deux modules (TE et MV) sont utilisés afin de rendre les cafards artificiels (messages) interprétables par machine comme présenté dans les sections précédentes.

ii. **Sélection du numéro d'abri :** Dans le problème du filtrage du spam, nous avons deux abris (spam et ham)

iii. **Initialisation de l'abri :** Au départ, l'abri ham et spam contient les emails (spam ou ham) de la base d'apprentissage

iv. **règles de dissimulation:** Un ensemble de règles de dissimulation (shelter darkness and security quality)) est utilisé par le cafard (message à classer) pour se déplacer afin d'être caché (classé) dans l'abri (classe) spam ou ham où il se sent plus en sécurité (classe idéale).

5.2.1.1- Opérateur Shelter Darkness (SD): Cet opérateur est basé sur l'élément de référence de chaque abri. Dans le problème du filtrage du spam, un message est attiré par la classe où il y a plus de ses messages similaires issus de la base d'apprentissage. L'obscurité de chaque abri (classe) est référencée par les messages initiaux placés dans chaque abri.

$$SD\left(S_{Spam}\right) = \frac{\mathrm{LB}\left(MS_{Spam}\right)}{\#\mathrm{MLB}}$$

$$SD\left(S_{Ham}\right) = \frac{\mathrm{LB}\left(MS_{Ham}\right)}{\#\mathrm{MLB}}$$

***#MLB*:** Le nombre total de base d'apprentissage des messages.

***LB (MS_{spam})* :** Le nombre de messages de spam dans la base d'apprentissage

***LB (MS_{ham})* :** Le nombre de messages ham dans la base d'apprentissage.

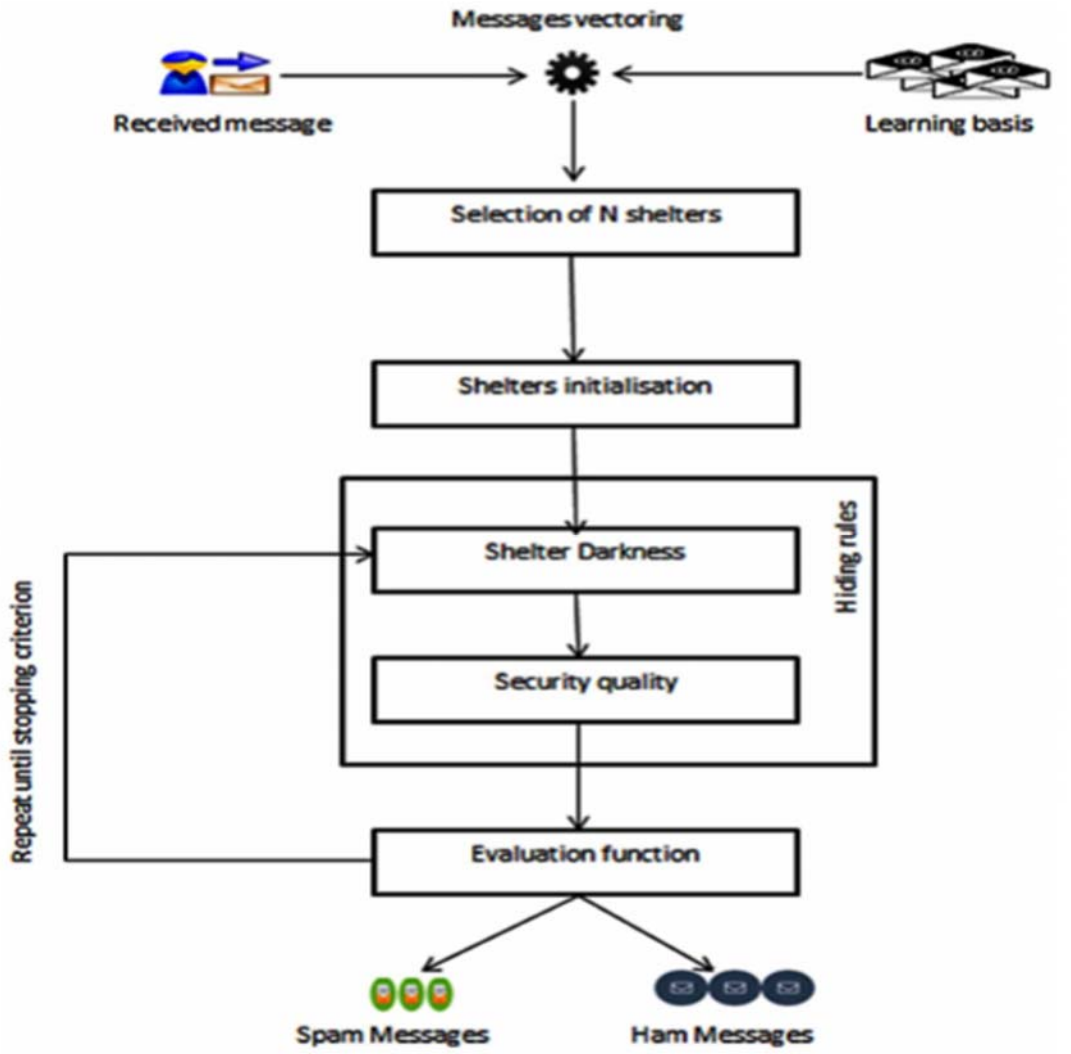

Figure 5 : Filtrage du spam basé sur la technique ASR.

5.2.1.2- Opérateur Security quality (SQ) : un message a une qualité de sécurité maximale s'il est proche du centroïde de l'abrit. La qualité de sécurité *SQ(Mn ,Si)* représente la distance entre le message à classer *Mn* et le centroïde de chaque classe.

$$SQ\left(M_n, S_i\right) = \frac{1}{\text{distance}\left(M_n, CS_i\right)}$$

CSi : Centroid of the shelter *Si*

5.2.1.3- Fonction d'évaluation (Attraction des abris)

Le message reçu sera classé dans la classe la plus attractive (abri) avec la fonction d'évaluation (SA) maximum.

$$SA\left(M_N, S_i\right) = \alpha * SD\left(M_N, S_i\right) + \beta * SQ\left(M_N, S_i\right)$$

Où :

M_N : Nouveau numéro email *i* (à classer) reçu par l'utilisateur.

S_i : Numéro de classe i.

α, β : Paramètres permettant d'ajuster l'importance de chaque règle.

5.2.1.4- Étape de mise à jour

Itérativement, les règles de dissimulation de chaque message seront mises à jour et les messages seront reclassés.

5.2.1.5- Critère d'arrêt

Le critère d'arrêt de l'algorithme ASC est soit le nombre d'itérations fixé à l'avance, soit si le Les messages dans chaque classe restent les mêmes pour l'itération i et l'itération i-1.

5.2.1.6 L'algorithme des cafards sociaux artificiels pour le filtrage du spam :

Le pseudo suivant résume l'algorithme des cafards sociaux artificiels pour le détection et filtrage des spam.

Input

Threshold, Distance measure

Weightings adjustment ʌ, β, α.

SMS spam V.0.1 dataset

While not stopping criterion do

For each no-secure cockroach Cn (test basis messages) do

For each shelter (S_{spam} and S_{ham}) do

Calculate

/*Shelter darkness*/

$$SD(Si) = \frac{LB(CSi)}{\#CLB}$$

/*Quality safety*/

$$SQ\ (Mn,Si) = \frac{1}{distance(Mn,CSi)}$$

/* the shelter attraction SA */

SA (Mn, Si) = α *SD (Mn, Si) +β*SQ (Mn, Si)

End

If SA (Mn, S_{spam})>SA (Mn, S_{ham}) then

Mn spam

Else

Mn ham

End

Updating the hiding rules

End

Return Spam class and ham class

Messages of each class.

5.2.1.7- Cartographie de la vie biologique à la vie artificielle de ASR :

Vie naturelle	Vie artificielle
Cafard	Instance
Un cafard fait le choix à partir de plusieurs abris et il choisis toujours l'abri le plus sécurisé pour ce cacher.	Chaque instance est classé dans la classe la plus approprier parmi un ensemble de class connu à l'avance.
Abri	Classe
Lorsque les cafards ne sentent pas bien ils quittent l'abri pour rechercher un autre abri plus sécurisé.	Mise à jour.
Les cafards positionnés au milieu de l'abri ont une qualité de sécurité plus élevée que les cafards position a la frontière de l'abri.	Les instances sont attirées par la classe avec le plus proche barycentre.
Chaque cafard est guidé par l'appel des représentants de chaque abri.	Chaque instance appartient à la classe avec les plus proches points de support (corrélation).
La base d'apprentissage est une information externe.	Obscurité de l'abri est une information externe.
Hésitation.	Probabilité d'hésitation.
Les risques	Algorithme naïve bayes.
Chaque cafard aura des obstacles et des risques d'être vu par un humain lors de son déplacement. Il choisit toujours la route ou il aura moins de risque.	Probabilité des risque : chaque instance Cn a une probabilité $\mathrm{P}\ (S_i/C_n)$ d'appartenir à une classe Si (chaque cafard a une probabilité P d'atteindre chaque abri) calculée par l'algorithme naïve bayes
Cafard est attirer par l'abri ou il y'a plus de cafards.	Instance est attirer par la classe qui contient plus d'instance de la base d'apprentissage.

Table 5– Cartographie de la vie biologique à la vie artificielle de ASR.

5.2.2. Algorithme 2 : Social Worker Bees (SWB) pour le filtrage du spam :

5.2.2.1. Algorithme Social Worker Bees (SWB)

Le SWB a été développé par Hamou & al in (Hamou, 2013) pour la détection des spams. C'est une technique basée sur la vie biologique des abeilles ouvrières (Colleteuses, gardienne, nettoyeuse) et comment elles assurent le bon fonctionnement de la ruche. Comme le montre la figure (5.1), dans notre travail, l'ensemble de données est divisé en deux parties: base d'évaluation et base d'apprentissage. Les SWB ont en entrée un message de la base d'évaluation (fleur) qui sera classé spam ou ham. Nous avons deux ruches (classes) spam et ham. Chaque message reçu doit passer par trois filtres (Manifold, Guardian et Cleaner). Chaque filtre doit calculer la distance moyenne entre le message M de la base d'évaluation (message à analyser) avec les messages de base d'apprentissage ham et avec les messages de base d'apprentissage spam. Chaque filtre est caractérisé par une mesure de distance et la distance utilisée par chaque filtre doit être différente des deux autres mesures utilisées par les autres filtres. Un message est ham si ses trois distances moyennes avec les messages de base d'apprentissage ham sont inférieures à la distance moyenne avec les messages de base d'apprentissage spam. Le rôle de chaque filtre est détaillé dans la suite:

Manifold Filter: L'abeille collecteur doit rechercher le bon miel qui existe dans la fleur et le recueillir. Dans cette étape, une distance moyenne sera calculée entre le message M (fleur) de la base d'évaluation avec les messages spam de la base d'apprentissage et la même distance est calculée avec les messages ham de la base d'apprentissage. Le message passe au filtre gardien si la distance moyenne du ham est inférieure à la distance moyenne du spam, sinon le message est classé comme spam.

Guardian Filter : Calcule la distance moyenne entre le message qui a passé le filtre collecteur et les messages de base d'apprentissage spam et la même distance est calculé avec le filtre de la base d'apprentissage ham. Si la distance moyenne du ham est inférieure à la distance moyenne du spam, le message passe au filtre suivant, sinon le message est classé comme spam.

Cleaner Filter : Pour ce filtre, le même processus que les deux filtres précédents mais la mesure de distance utilisée par chaque filtre doit être différente avec les deux autres filtres

Par exemple, les trois filtres sont (manifold filter utilise la distance cosinus, guardian filter utilise la distance Manhattan et cleaner filter utilise la distance euclidienne). Nous calculons la distance moyenne entre le nouveau message M de la base d'évaluation avec les messages de base d'apprentissage ham et avec le messages de base d'apprentissage spam. Le message M est classé ham si toutes les distances moyennes avec les messages de base d'apprentissage ham sont

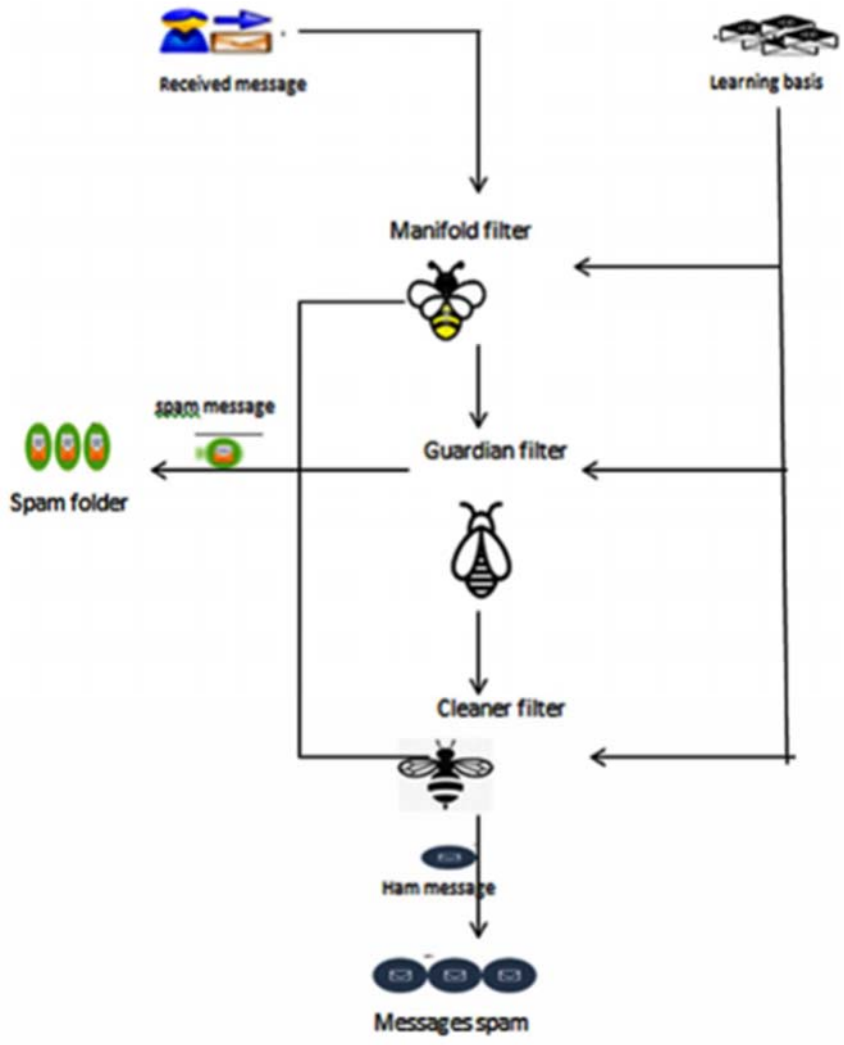

Figure (5.1). Filtrage du spam basé sur la technique SWB des abeilles assistantes sociales.

inférieures aux distances moyennes avec les messages de base d'apprentissage de spam.

Algorithme Social Worker Bees (SWB):

```
Input
   Spam learning basis, ham learning basis, Test basis
  Three distance measures
                    /* artificial inspiration */
 For each new message M_i do
     Filtering by manifold filter
      If message M is Ham then
        Filtering by guardian filter
         If message M is a Ham then
         Filtering by cleaner filter
         If message M is a Ham then
        The message M Ham
      Else the message M spam
 End Do
Return:
Folder (ruche) of Spam messages
 Folder (ruche) of Ham messages
```

5.2.2.2. Cartographie de la vie naturel à la vie artificielle de SWB :

Vie naturelle des abeilles travailleuses	**Classificateur non supervisés**
Ruche	Cluster
Abeille reine	Centroid
Abeille collecteuse	Filtre collecteur
Abeille gardienne	Filtre guardien
Abeille néttoyeuse	Filtre néttoyeuse
Communication	Fonction de fitness
Apiary	Ensemple de clusters
Assurer le bon fonctionnement de la ruche	Assure que les textes du méme cluster doivent être le plus similaire que possible et les texte de different cluster doivent être le plus dissimilaire que possible
Les fleurs	Ensemble de données
le miel	Les documents après la classification

Table 5.1– Cartographie de la vie biologique à la vie artificielle de SWB.

5.3. Architecture du système :

L'architecture de notre modèle de filtrage des courriels est représentée sur la figure 5.2. Tout d'abord, en utilisons le corpus SMS Spam Collection. Le corpus de messagerie va passer par la phase de préparation ou prétraitement (la représentation et le codage et la vectorisation de texte) puis en utilisant l'un des algorithmes d'apprentissage (SVM, NB, KNN) et les algorithmes bio inspiré proposé (ASR, SWBs) pour construit un modèle qui permet de classer les nouveaux messages.

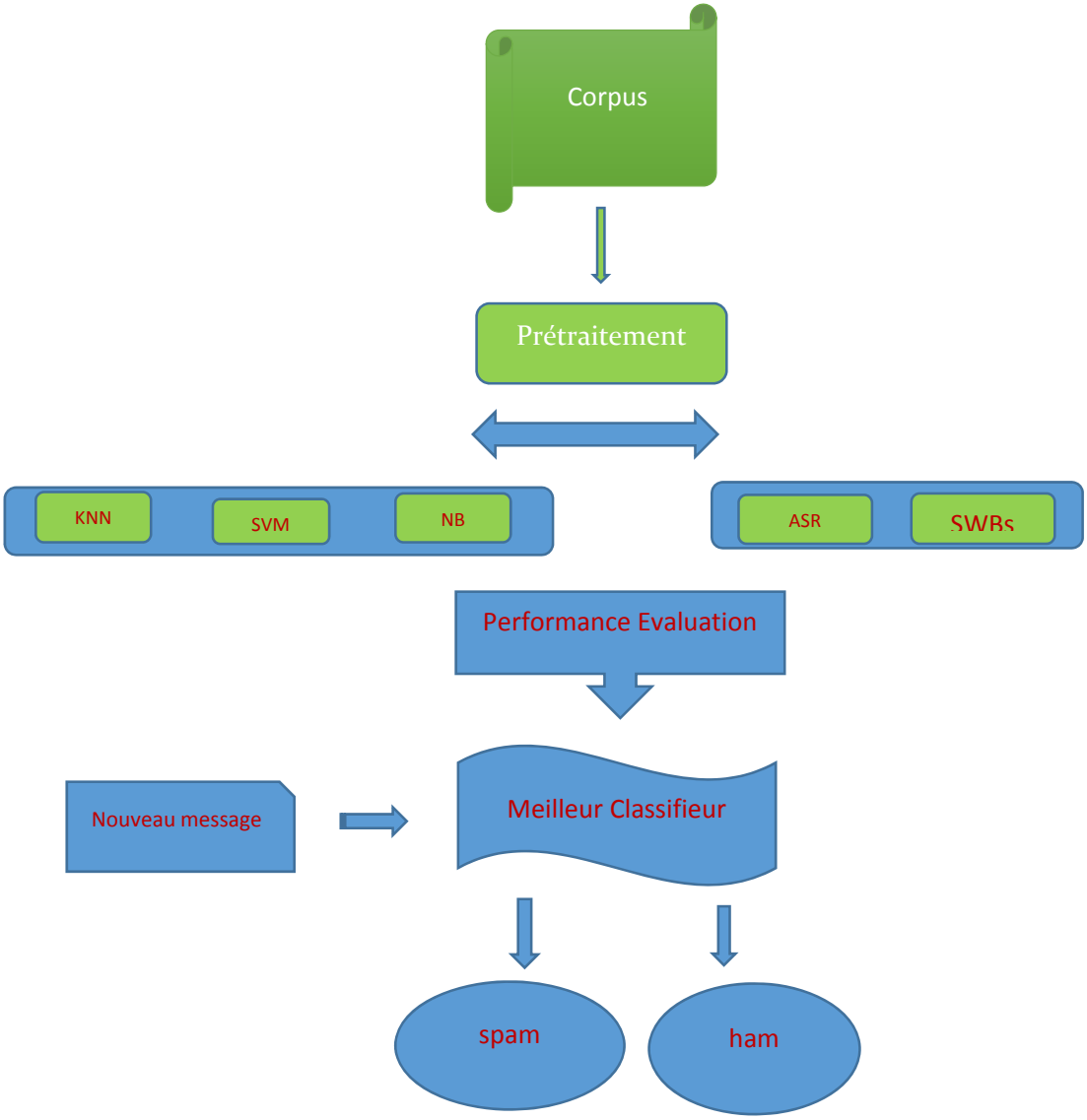

Figure 5.2 – Architecture du système

5.4. Méthode bio-inspire pour la détection des spam :

Á la suite des problèmes de détection des spam, nous avons utilisé des algorithmes inspirés du comportement de la vie social des cafards et des abeilles ouvrières (collecteuse, gardienne, nettoyeuse). Dans ce chapitre nous traitons le problème de détection des spam avec ces algorithmes de la vie naturel en utilisant le corpus SMS Spam Collection Data Set pour l'expérimentation et les résultats obtenus seront comparés avec d'autres classificateurs comme (SVM, KNN, NB).

5.5. Prétraitement des données textuelles :

5.5.1. Représentation de texte :

Dans cette représentation, les termes sont des mots qui constituent un texte. Dans les langues comme le français ou l'anglais et autres langages, les mots sont séparés par des espaces ou des signes de ponctuations ; ces derniers, tout comme les chiffres, sont supprimés de la représentation. Les composantes des vecteurs peuvent être une fonction de l'occurrence des mots dans le texte. Cette représentation exclue toute analyse grammaticale et toute notion de distance entre les mots, et c'est pour quoi elle est appelée « sac de mots » et « caractère N-Gram»

Présentation de texte basé sur la transformation de chaque texte en une liste de petits unités, appelées termes (terme peut être un ensemble de caractères, un mot, une phrase, un concept, etc.), selon la méthode utilisée et les résultats que nous voulons atteindre

A. **Sac de mots :** Cette méthode est simple et la plus ancienne basée sur la division du texte en un ensemble de mots (un mot est un ensemble de caractères liés entre eux séparés par des ponctuations) plusieurs inconvénients sont été déterminés à propos de cette méthode considère l'ambiguïté du langage naturel
B. **Caractère N-Gram :** Il est basé sur un paramètre N qui représente une fenêtre de caractères, qui ne se déplacent pas à pas dans le texte et enregistrent pour chaque mouvement la fenêtre de capture dans liste, cette technique présente plusieurs avantagent.

5.5.2. Codage de texte :

En calcule la fréquence de chaque attribut (composant) dans chaque document en utilisant une pondération liée au texte lui-même (ex: le nombre d'occurrences d'un terme dans le texte) et à l'ensemble de données en intégralité (ex: le nombre d'occurrences du terme dans l'ensemble des données).

5.5.3. Déférentes techniques de pondération :

5.5.3.1 La pondération TF :

La Fréquence du terme dans le document (nombre d'occurrences)

Elle possède quelques avantages comme :

- On capte plus d'information, la répétition d'un terme dans le document est prise en compte
- Des techniques savent prendre en compte ce type d'information (calcul matriciel)

Et des Inconvénients :

- Les écarts entre documents sont exagérés (ex.si on utilise une distance euclidienne).
- On ne tient pas compte de la prévalence des termes dans l'ensemble des documents (cf.IDF)

5.5.3.2. La pondération TF-IDF:

La fonction Tf-IDF (acronyme pour « term frequency inverse document frequency ») est la pondération la plus utilisée dans la littérature. Sa force réside dans le fait qu'elle implémente en même temps : l'Exhaustivité et Spécificité.

Le poids de terme tk appartenant au document di égale à:

$$\text{TF} - \text{IDF (tk, di)} = \text{N} * \log\frac{A}{B}$$

- **N:** le nombre d'occurrences du terme tk dans le texte di
- **A:** le nombre total de textes du corpus;
- **B:** le nombre de textes dans lesquels le terme tk apparaît au moins une fois.

5.5.3.3. Vectorisation de texte :

La transformation de chaque texte en un vecteur où chaque composant les vecteurs représentent la position et l'importance de chaque terme et le corpus sera représenté par un document matriciel * terme.

5.6. Implémentation et résultats :

5.6.1. Environnement et outils de développement :

Langage JAVA : Notre choix du langage de programmation s'est porté sur le langage JAVA. Java est un langage de programmation et une plateforme informatique qui ont été créés par Sun Microsystems en 1995. JAVA est un langage orienté objet simple ce qui réduit les risques d'incohérence et il possède une riche bibliothèque de classes comprenant des fonctions diverses telles que les fonctions standards, le système de gestion de fichiers ainsi que beaucoup de fonctionnalités qui peuvent être utilisé pour développer des applications diverses. Aussi, il offre un nombre important de fonctions de traitement de texte. Java est rapide, sécurisé et fiable. La technologie Java est présente sur tous les fronts

Eclipse : est un projet, décliné et organisé en un ensemble de sous-projet de développement logiciel, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent lancé par IBM. Son objectif (avantage) et de produire et fournir des outils pour la réalisation de logiciels englobant les activités programmation et fournir aussi un environnement intégré qui facilite la création et la compilation, les teste et l'exécution de projets.

Weka : Nous avons effectué des expériences en intégrant quelque bibliothèque de l'outil WEKA dans notre IDE, en fait, (Waikato Environnement Knowledge Analysis) offrent un ensemble des algorithmes permettant de manipuler et d'analyser des fichiers de donnés. Il se compose principalement :

- De classe Java permettant de charger et manipuler des donnés.
- De classe Java pour implémenter les principaux algorithmes de classification supervisée et non supervisée.
- D'outils de sélection d'attributs, des statistiques sur ces attributs.
- De classes permettant de visualiser les résultats.

Figure 5.3 –Weka GUI.

Objectifs :

- Le weka explorer permet de lancer une méthode à partir d'un fichier ARFF
- Les résultats sont mis sous la forme d'un fichier texte normaliser
- Permet de sélectionner la méthode (l'algorithme) la mieux adaptée ou la plus efficace

5.6.2. Description du corpus utilisé :

Le corpus SMS Spam Collection est un ensemble commun de messages étiquetés SMS. Il dispose d'une collection composée de 5 574 messages en anglais, réels, étiqueté selon étant légitime (Ham) ou spam. Cette collection contient 747 messages spam et 4827 messages légitimes.

Utilisation : La collection est composée d'un seul fichier texte où chaque ligne contient le message brut suivie par la bonne classe. Nous vous proposons quelques exemples ci-dessous :

```
What you doing? how are you? ham

Ok lar... Joking wif u oni... ham

dun say so early hor... U c already then say... ham

MY NO. IN LUTON 0125698789 RING ME IF UR AROUND! H* ham

Siva is in hostel aha:-. ham

Cos i was out shopping wif darren jus now n i called him 2 ask wat
present he wan lor. Then he started guessing who i was wif n he
finally guessed darren lor. ham

FreeMsg: Txt: CALL to No: 86888 & claim your reward of 3 hours talk
time to use from your phone now! ubscribe6GBP/ mnth inc 3hrs 16
stop?txtStop spam

Sunshine Quiz! Win a super Sony DVD recorder if you canname the
capital of Australia? Text MQUIZ to 82277. B spam

URGENT! Your Mobile No 07808726822 was awarded a L2,000 Bonus Caller
Prize on 02/09/03! This is our 2nd attempt to contact YOU! Call
0871-872-9758 BOX95QU spam
```

Figure 5.4 corpus SMS Spam Collection.

5.6.3. Les mesures d'évaluation :

Les critères de mesure des performances sont le rappel et la précision et aussi la f-mesure qui est la combinaison des deux précédentes. Pour mesurer toutes ces métriques, nous devons tout d'abord calculer les valeurs suivantes:

- Vrai négatifs (VN) : le nombre de courriels spam classifiés spam.
- Vrai positifs (VP) : le nombre de courriels légitimes classifiés légitimes
- Faux Positifs (FP) : le nombre de courriels spam classifiés légitimes.
- Faux négatifs (FN) : le nombre de courriels légitimes classifiés spam

Matrice de contingence		Jugement de l'expert	
		Vrais	Faux
Jugement de notre système	Vrais	VP_i	FP_i
	Faux	FN_i	VN_i
Vrais positive (VP):	Le nombre d'instances attribués à une catégorie convenablement. (instances attribués à leurs vraies catégories)		
Vrais Négative (VN):	Le nombre d'instances non attribués à une catégorie convenablement (Qui n'ont pas à être attribués à une catégorie, et ne l'ont pas été)		
Faux positive (FP):	Le nombre d'instances attribués à une catégorie inconvenablement. (instances attribués à des mauvaises catégories)		
Faux négative (FN):	Le nombre d'instances inconvenablement non attribués. (Qui auraient dû être attribués à une catégorie mais qui ne l'ont pas été)		

Figure 5.5 matrice de confusion.

Nous avons alors les mesures suivantes :

Rappel (R) : Le rappel mesure la capacité de notre système à détecter les instances bien classées. Le R représente le rapport entre le nombre de documents correctement classés par notre système dans la classe Ci par rapport au nombre total des documents réellement dans la classe c.

$$\mathbf{R} = \frac{\boldsymbol{VPi}}{\boldsymbol{VPi} * \boldsymbol{FNi}}$$

Précision (P) : La précision permet de mesurer la capacité d'un système à retourner seulement les instances bien classées. La P représente le rapport entre le nombre d'instances correctement classé par l'algorithme dans la classe Ci par rapport au nombre d'instances classées par notre système dans la classe Ci.

$$\mathbf{P} = \frac{\boldsymbol{VPi}}{\boldsymbol{VPi} * \boldsymbol{FPi}}$$

F-mesure (F) : La f-mesure permet de calculer la qualité de classification d'un algorithme à partir du rappel et de la précision.

$$\boldsymbol{F - Measure} = \frac{\boldsymbol{2 * précision * rappel}}{\boldsymbol{précision * rappel}}$$

5.6.4. Description du système :

Dans cette partie nous nous intéressons à la présentation de notre système en montrant quelques exemples de captures d'écran des différentes interfaces réalisées. Nous avons essayé de créer une interface graphique qui montre le plus possible les détails d'exécution de notre application.

Figure 5.6 – L'interface de l'application.

Ici nous remarquons ce qu'il y a dans paramètre en trouve quatre listes de choix pour trouver des résultats de l'expérimentation :

Figure 5.7 – listes de choix pour trouver des résultats de l'expérimentation.

Choix des options Prétraitement, Pondération et méthode de validation avec l'algorithme qui peut utiliser l'un des deux algorithmes bio-inspirer ASR et SWBs :

- Prétraitement : on a deux choix soit avec les mots vides ou aans les mots vides.
- Pondération: en calcule la pondération de chaque token avec l'un des méthodes : TF ou TF-IDF.
- Méthodes de validations : on a deux chois soit K-Fold (Validation croisée à 10) ou training and Test Set.

5.6.5. Résultats de l'expérimentation :

5.6.5.1. Avec l'algorithme ASR :

Résultat 1 obtenus en utilisant la représentation par sac de mots ; pondération TF, Prétraitement Avec les mots vide et méthodes de validations K-Fold (validations croisées à 10 plis) :

Fold	rappel	précision	f-mesure	TS
Fold n 0	0.580	0.666	0.517	0.580
Fold n 1	0.784	0.836	0.776	0.784
Fold n 2	0.745	0.815	0.730	0.745
Fold n 3	0.775	0.845	0.762	0.775
Fold n 4	0.830	0.863	0.826	0.830
Fold n 5	0.810	0.851	0.804	0.810
Fold n 6	0.824	0.870	0.818	0.824
Fold n 7	0.853	0.886	0.850	0.853
Fold n 8	0.840	0.879	0.836	0.840
Fold n 9	0.800	0.845	0.793	0.800

Table 5.2–résultat Numéro 01.

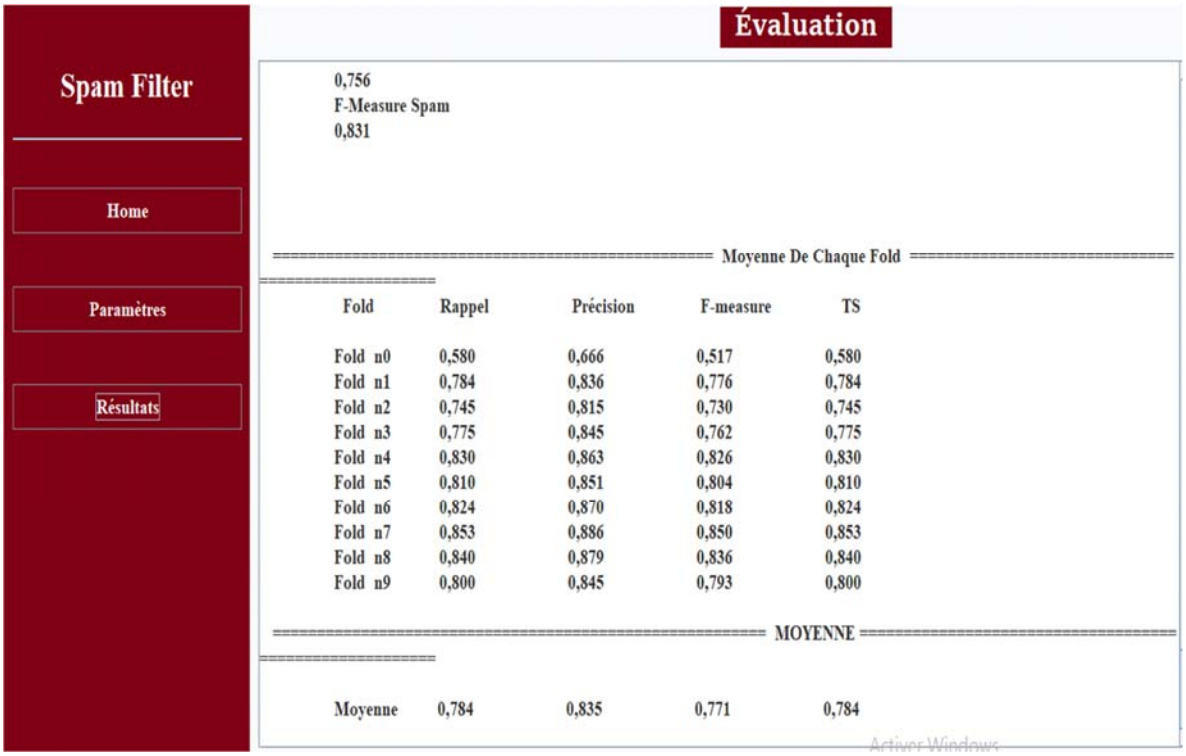

Figure 5.8 – résultat Numéro 01.

Résultat 2 obtenus en utilisant la représentation par sac de mots ; pondération TF, Prétraitement sans les mots vide et méthodes de validations K-Fold (validations croisées à 10 plis) :

Fold	rappel	précision	f-mesure	0.540
Fold n 0	0.540	0.568	0.488	0.784
Fold n 1	0.784	0.836	0.766	0.725
Fold n 2	0.725	0.823	0.703	0.784
Fold n 3	0.784	0.836	0.776	0.830
Fold n 4	0.830	0.863	0.826	0.800
Fold n 5	0.800	0.857	0.792	0.804
Fold n 6	0.804	0.837	0.799	0.833
Fold n 7	0.833	0.856	0.831	0.800
Fold n 8	0.800	0.857	0.792	0.790
Fold n 9	0.790	0.828	0.784	0.540

Table 5.3–résultat Numéro 02.

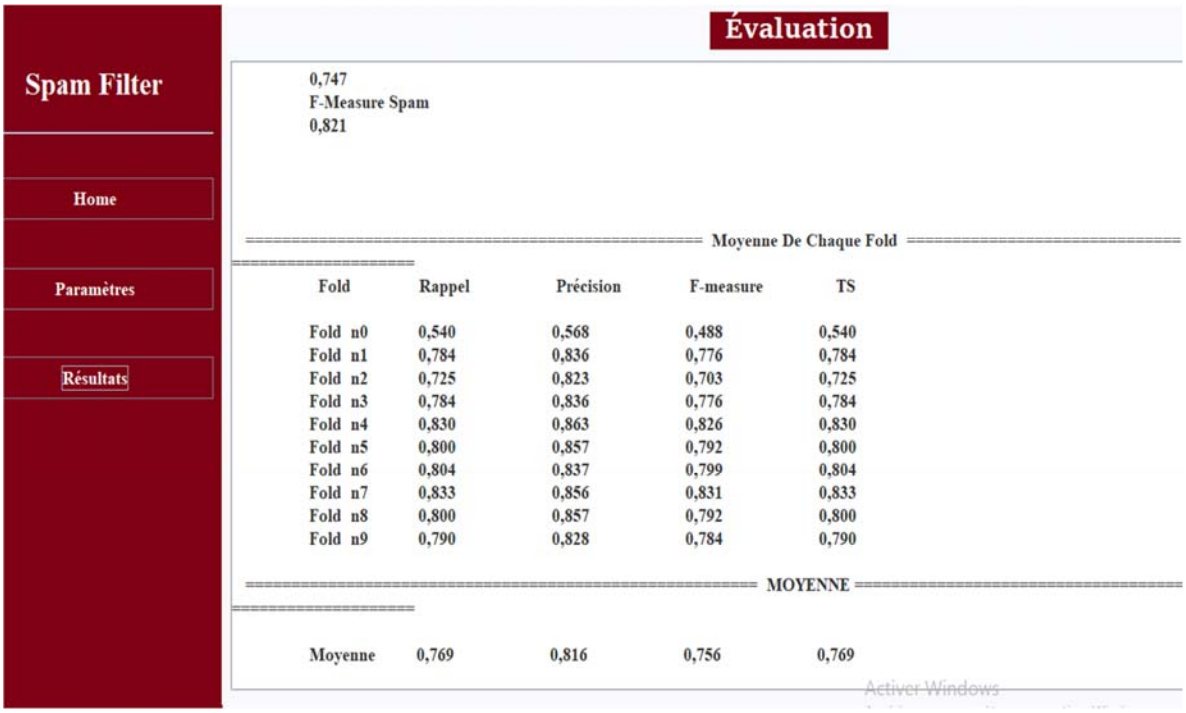

Figure 5.9 – résultat Numéro 02.

Résultat 3 obtenus en utilisant la représentation par sac de mots ; pondération TF-IDF, Prétraitement sans les mots vide et méthodes de validations K-Fold (validations croisées a 10 plis) :

Fold	rappel	précision	f-mesure	TS
Fold n 0	0.830	0.873	0.825	0.830
Fold n 1	0.850	0.885	0.847	0.850
Fold n 2	0.840	0.879	0.836	0.840
Fold n 3	0.850	0.885	0.847	0.850
Fold n 4	0.775	0.845	0.762	0.775
Fold n 5	0.760	0.838	0.745	0.760
Fold n 6	0.780	0.847	0.769	0.780
Fold n 7	0.780	0.833	0.771	0.780
Fold n 8	0.780	0.847	0.769	0.780
Fold n 9	0.780	0.847	0.769	0.780

Table 5.4–résultat Numéro 03.

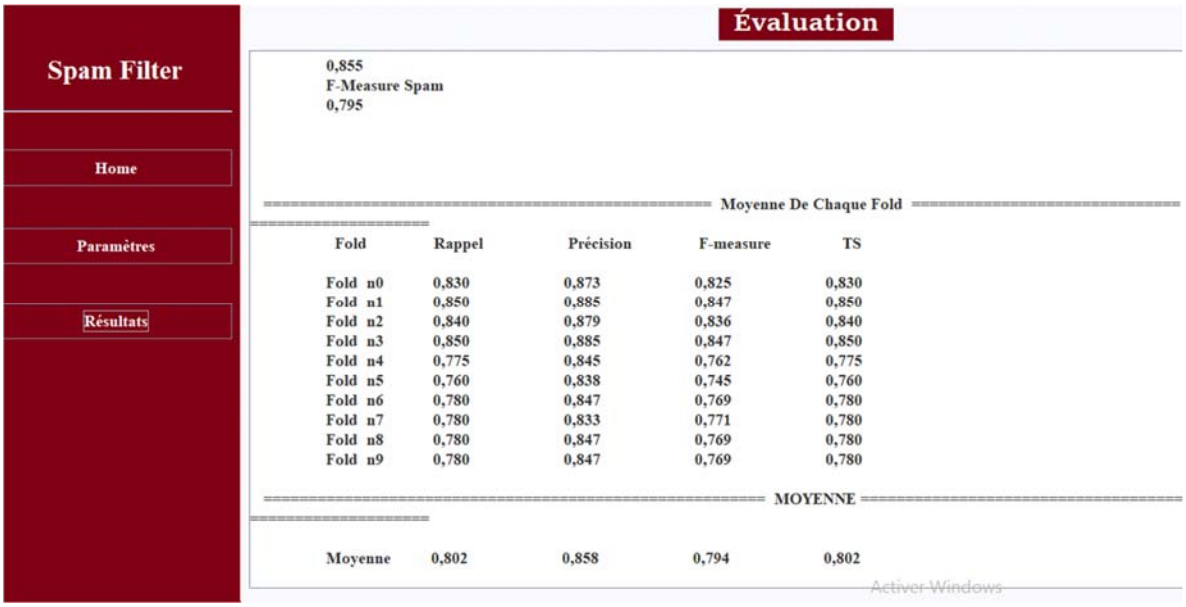

Figure 5.10 – résultat Numéro 03.

Résultat 4 obtenus en utilisant la représentation par sac de mots ; pondération TF-IDF, Prétraitement avec les mots vide et méthodes de validations K-Fold (validations croisées a 10 plis) :

Fold	rappel	précision	f-mesure	TS
Fold n 0	0.790	0.852	0.780	0.790
Fold n 1	0.790	0.852	0.780	0.790
Fold n 2	0.820	0.868	0.814	0.820
Fold n 3	0.800	0.857	0.792	0.800
Fold n 4	0.794	0.854	0.785	0.794
Fold n 5	0.830	0.873	0.825	0.830
Fold n 6	0.830	0.873	0.825	0.830
Fold n 7	0.770	0.828	0.759	0.770
Fold n 8	0.800	0.857	0.792	0.800
Fold n 9	0.840	0.879	0.836	0.840

Table 5.5–résultat Numéro 04.

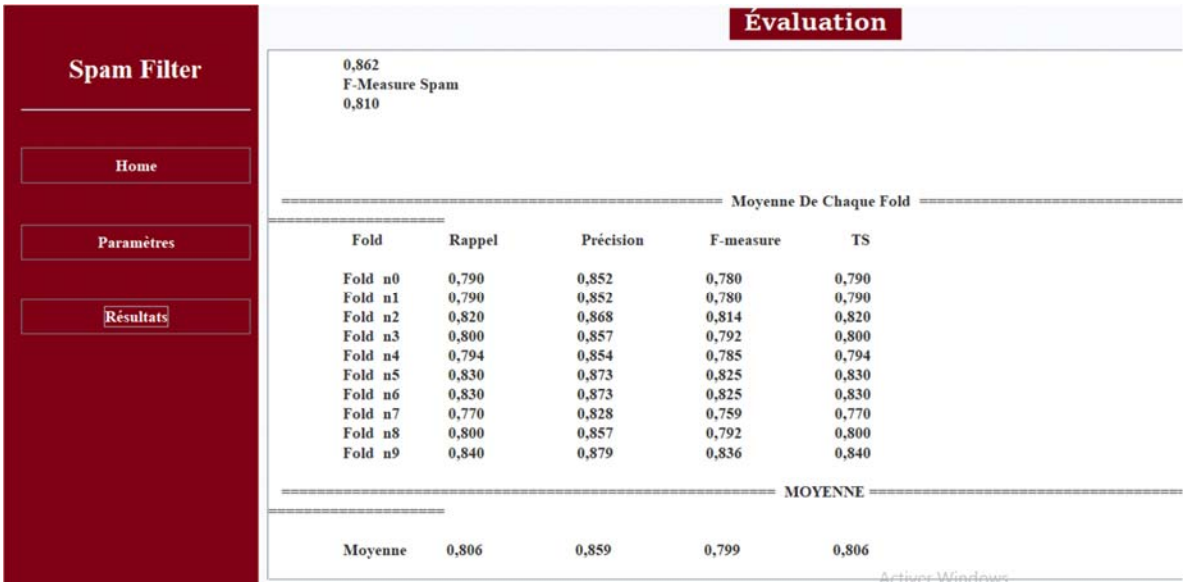

Figure 5.11 – résultat Numéro 04.

Résultat 5 obtenus en utilisant la représentation par sac de mots ; pondération TF IDF, Prétraitement Avec les mots vide et méthodes de validations Training and Test Set :

Rappel Ham	Rappel spam	précision Ham	précision Ham	f-mesure Ham	f-mesure Ham
0.706	0.691	0.947	0.766	0.809	0.852

Table 5.6–résultat Numéro 05.

moyenne	rappel	précision	f-mesure	TS
valeur	0.840	0.879	0.836	0.840

Table 5.7–résultat Numéro 05.

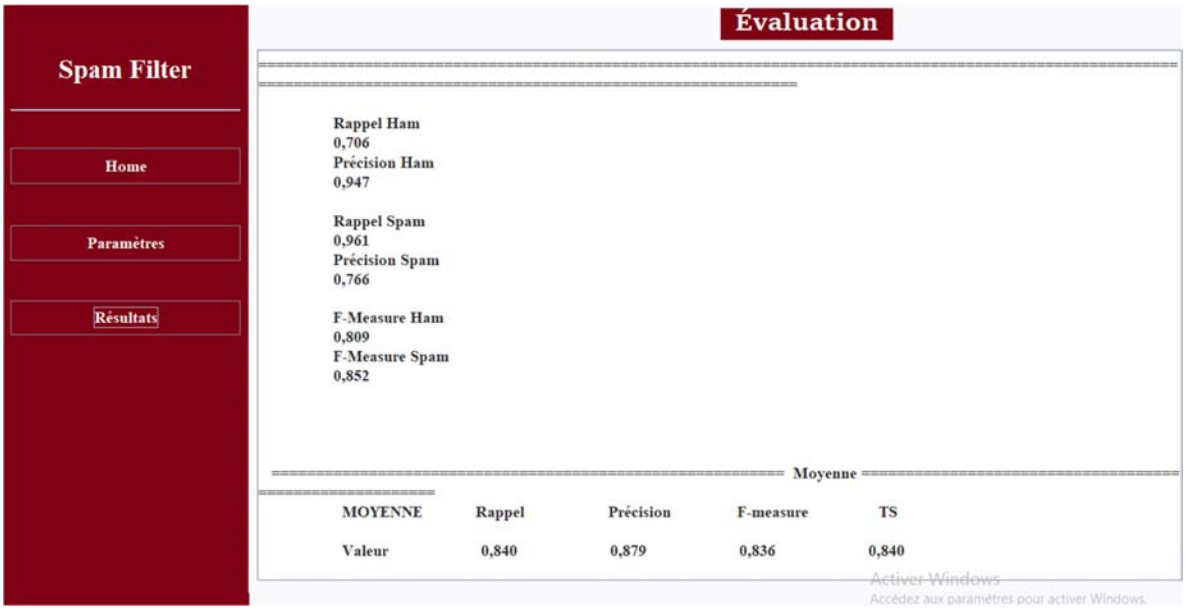

Figure 5.12 – résultat Numéro 05.

5.6.5.2. Avec l'algorithme SWBs:

Résultat 6 obtenus en utilisant la représentation par sac de mots ; pondération TF, Prétraitement Avec les mots vide et méthodes de validations K-Fold (validations croisées à 10 plis) :

Fold	rappel	précision	f-mesure	TS
Fold n 0	0.810	0.851	0.804	0.810
Fold n 1	0.784	0.836	0.776	0.784
Fold n 2	0.784	0.836	0.776	0.784
Fold n 3	0.745	0.815	0.730	0.745
Fold n 4	0.830	0.841	0.829	0.830
Fold n 5	0.814	0.864	0.807	0.814
Fold n 6	0.804	0.847	0.798	0.804
Fold n 7	0.824	0.832	0.822	0.824
Fold n 8	0.824	0.836	0.822	0.824
Fold n 9	0.720	0.786	0.703	0.720

Table 5.8–résultat Numéro 06.

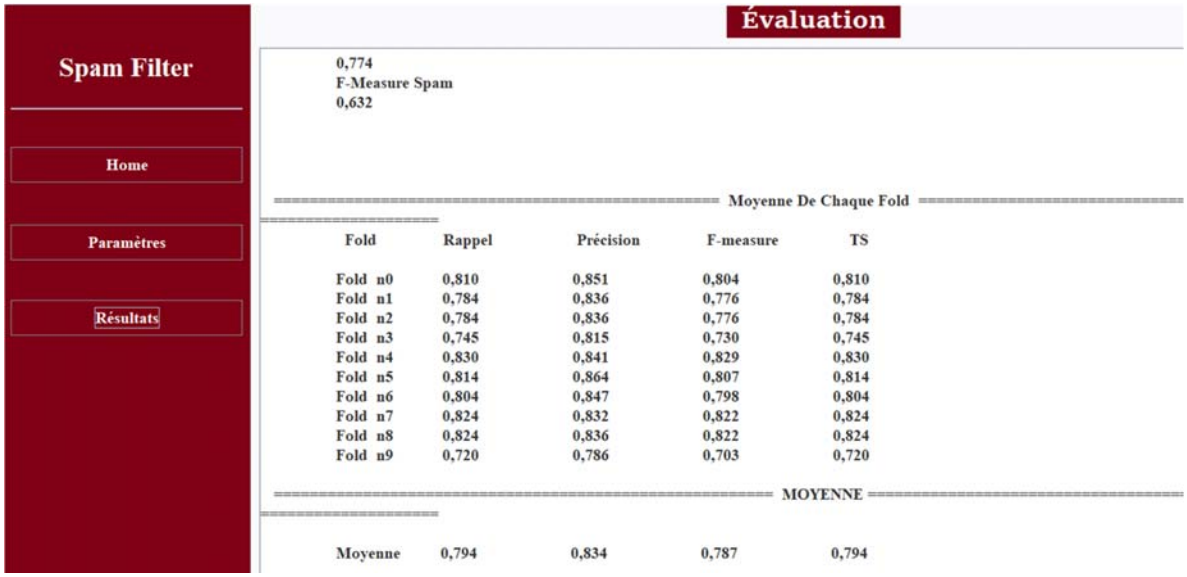

Figure 5.13 – résultat Numéro 06.

Résultat 7 obtenus en utilisant la représentation par sac de mots ; pondération TF, Prétraitement sans les mots vide et méthodes de validations K-Fold (validations croisées à 10 plis) :

Fold	rappel	précision	f-mesure	TS
Fold n 0	0.800	0.857	0.792	0.800
Fold n 1	0.775	0.845	0.762	0.775
Fold n 2	0.775	0.831	0.765	0.775
Fold n 3	0.784	0.836	0.776	0.784
Fold n 4	0.800	0.857	0.792	0.800
Fold n 5	0.770	0.821	0.799	0.770
Fold n 6	0.784	0.836	0.776	0.784
Fold n 7	0.725	0.823	0.703	0.725
Fold n 8	0.775	0.845	0.762	0.775
Fold n 9	0.700	0.812	0.670	0.700

Table 5.9–résultat Numéro 07.

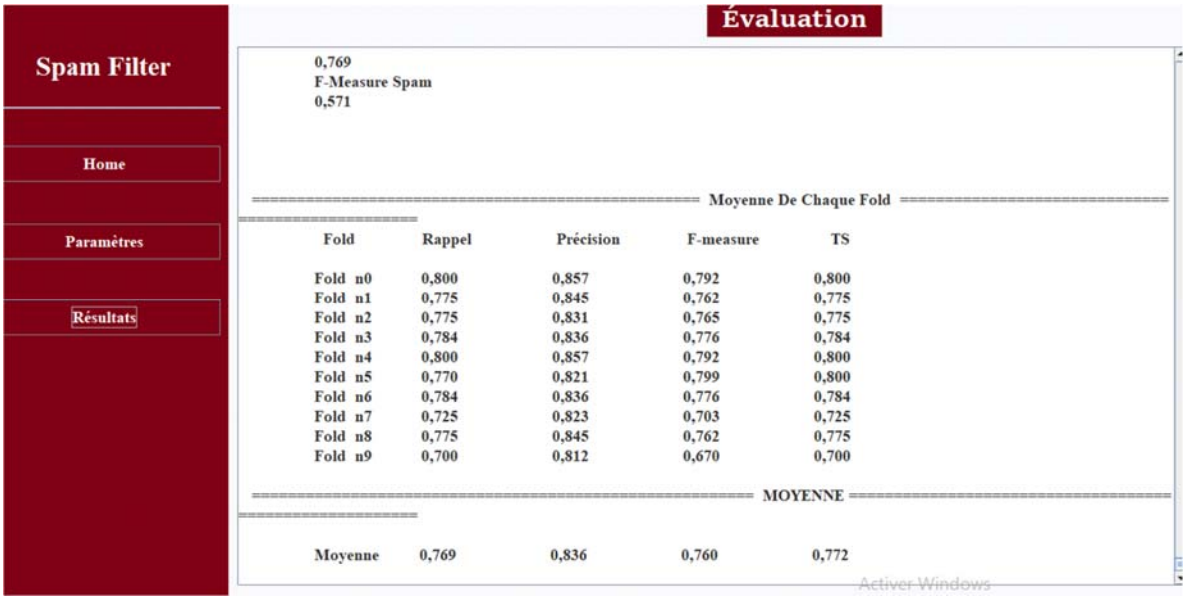

Figure 5.14 – résultat Numéro 07.

Résultat 8 obtenus en utilisant la représentation par sac de mots ; pondération TF-IDF, Prétraitement sans les mots vide et méthodes de validations K-Fold (validations croisées a 10 plis) :

fold	rappel	précision	f-mesure	TS
Fold n 0	0.780	0.847	0.769	0.780
Fold n 1	0.725	0.823	0.703	0.725
Fold n 2	0.814	0.864	0.807	0.814
Fold n 3	0.775	0.831	0.765	0.775
Fold n 4	0.784	0.836	0.776	0.784
Fold n 5	0.800	0.857	0.792	0.800
Fold n 6	0.780	0.847	0.769	0.780
Fold n 7	0.780	0.833	0.771	0.780
Fold n 8	0.780	0.847	0.769	0.780
Fold n 9	0.794	0.854	0.785	0.794

Table 5.10–résultat Numéro 08.

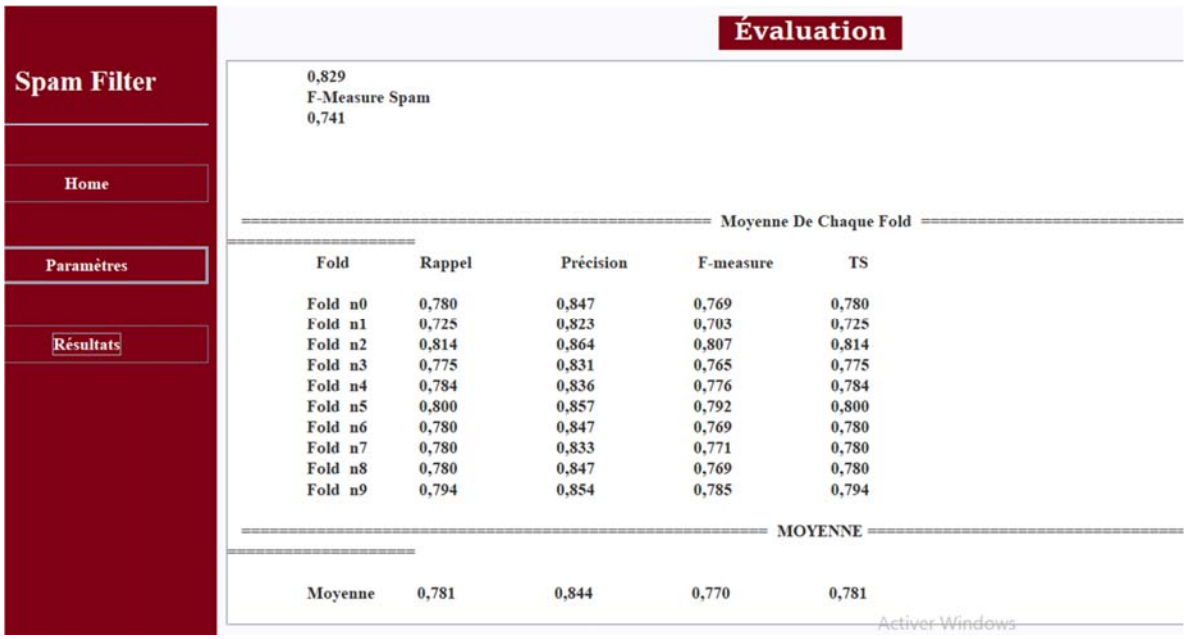

Figure 5.15 – résultat Numéro 08.

Résultat 9 obtenus en utilisant la représentation par sac de mots ; pondération TF-IDF, Prétraitement avec les mots vide et méthodes de validations K-Fold (validations croisées a 10 plis) :

Fold	rappel	précision	f-mesure	TS
Fold n 0	0.790	0.852	0.780	0.790
Fold n 1	0.790	0.852	0.780	0.790
Fold n 2	0.820	0.868	0.814	0.820
Fold n 3	0.760	0.838	0.745	0.760
Fold n 4	0.780	0.847	0.769	0.780
Fold n 5	0.780	0.833	0.771	0.780
Fold n 6	0.930	0.873	0.825	0.930
Fold n 7	0.770	0.828	0.759	0.770
Fold n 8	0.800	0.857	0.792	0.800
Fold n 9	0.760	0.838	0.745	0.760

Table 5.11–résultat Numéro 09.

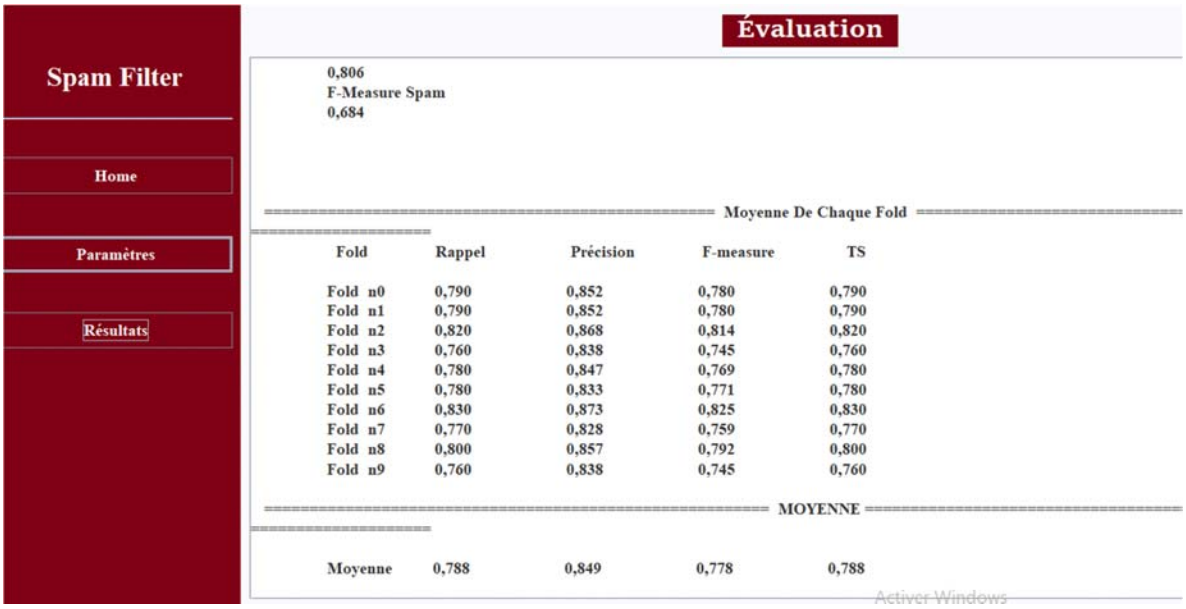

Figure 5.16 – résultat Numéro 09.

Résultat 10 obtenus en utilisant la représentation par sac de mots ; pondération TF IDF, Prétraitement Avec les mots vide et méthodes de validations Training and Test Set :

Rappel Ham	Rappel spam	précision Ham	Précision spam	f-mesure Ham	f-mesure spam
1.000	0.660	0.746	0.1.00	0.855	0.795

Table 5.12–résultat Numéro 10.

moyenne	**rappel**	**précision**	**f-mesure**	**TS**
Valeur	0.830	0.873	0.825	0.830

Table 5.13–résultat Numéro 11.

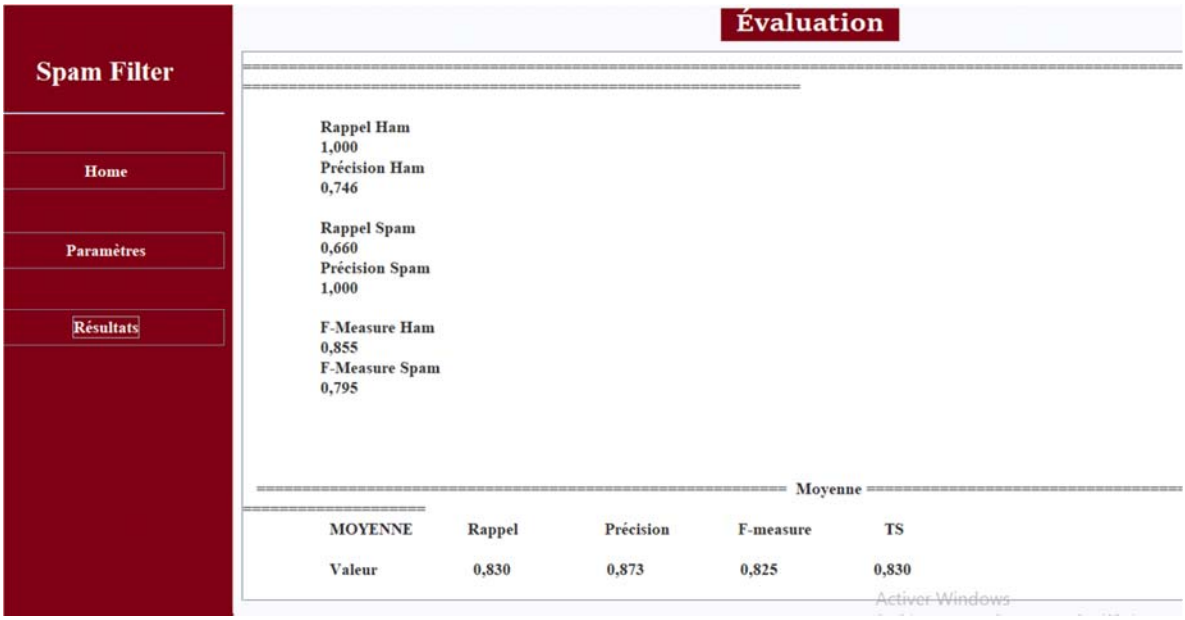

Figure 5.17 – résultat Numéro 10.

5.6.5.3. Résultats sur Weka :

Résultats obtenus par Naïve Bayes :

```
Time taken to build model: 0.14 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        5491               98.5109 %
Incorrectly Classified Instances        83                1.4891 %
Kappa statistic                          0.9355
Mean absolute error                      0.0215
Root mean squared error                  0.1142
Relative absolute error                  9.2547 %
Root relative squared error             33.5188 %
Total Number of Instances             5574

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,992     0,062     0,990       0,992   0,991      0,936    0,984     0,996     0
               0,938     0,008     0,950       0,938   0,944      0,936    0,984     0,970     1
Weighted Avg.  0,985     0,054     0,985       0,985   0,985      0,936    0,984     0,993

=== Confusion Matrix ===

    a    b   <-- classified as
 4790   37 |    a = 0
   46  701 |    b = 1
```

Figure 5.18 – Résultats obtenus par Naïve Baye.

Résultats obtenus par KNN :

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         866               69.4467 %
Incorrectly Classified Instances       381               30.5533 %
Kappa statistic                          0.289
Mean absolute error                      0.3053
Root mean squared error                  0.5252
Relative absolute error                 63.5591 %
Root relative squared error            107.1729 %
Total Number of Instances             1247

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,300    0,041    0,829      0,300    0,441      0,360    0,557     0,607     ham
               0,959    0,700    0,672      0,959    0,790      0,360    0,557     0,659     spam
Weighted Avg.  0,694    0,436    0,735      0,694    0,650      0,360    0,557     0,638

=== Confusion Matrix ===

   a   b   <-- classified as
 150 350 |   a = ham
  31 716 |   b = spam
```

Figure 5.19 – Résultats obtenus par KNN.

Résultats obtenus par SVM :

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1117               89.575  %
Incorrectly Classified Instances       130               10.425  %
Kappa statistic                          0.7846
Mean absolute error                      0.1328
Root mean squared error                  0.3009
Relative absolute error                 27.6347 %
Root relative squared error             61.3882 %
Total Number of Instances             1247

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,892    0,102    0,854      0,892    0,873      0,785    0,926     0,847     ham
               0,898    0,108    0,926      0,898    0,912      0,785    0,926     0,945     spam
Weighted Avg.  0,896    0,105    0,897      0,896    0,896      0,785    0,926     0,906

=== Confusion Matrix ===

   a   b   <-- classified as
 446  54 |   a = ham
  76 671 |   b = spam
```

Figure 5.20 – Résultats obtenus par SVM.

5.6.5.4. Comparaison et discussion de résultats :

Algorithme	Rappel	Précision	F-Mesure	TS
Classificateur ASR	0.806	**0.859**	**0.799**	**0.806**
Classificateur SWBs	**0.794**	**0.834**	**0.787**	**0.794**
KNN	**0.694**	**0.765**	**0.650**	**0.694**
Naive bayes	**0.927**	**0.929**	**0.927**	**0.927**
SVM	**0.896**	**0.897**	**0.896**	**0.895**

Table 5.14–résultat de Comparaison.

Après une étude comparative entre les classificateurs bioinspirés et les classificateurs classiques d'apprentissage machine (KNN, SVM et NB) en utilisant Weka , les résultats obtenus montrent une remarquable réussite des classificateurs SWBs et ASR qui est inspiré de la vie biologique des cafards et abeilles avec un taux de succès de 80% pour (ASR) et de 79% pour (SWBs) des instances correctement classées dans la validation croisée par contre dans Training Test on a trouvé 84% pour (ASR) et 83% pour(SWBs) des instances correctement classées.

D'après les résultats, le meilleur classificateur est le classificateur par Naïve bayes qui a donné une précision de 0.929 par rapport à notre classificateurs ASR ; SWBs 0.859 ; 0.834 de précision et même par rapports aux autres algorithmes (KNN et SVM).

Conclusion général

Conclusion général :

De notre vie normale, il y a beaucoup de chose qui peuvent être déduites et utiliser pour le bénéfice de l'humanité dans la technologie et la science, et leur utilisation qui nous permet de résoudre des problèmes et donner des bons résultats. Plusieurs recherches et applications nées de cette démarche en pleine expansion : le bio mimétisme

Comment faire face à des problèmes complexes en inspirant de cette nature et comment traduire le processus naturel (réel) en un processus artificiel (développer) ? Plusieurs travaux basés sur cette inspiration ont été réalisés dans des domaines déférents. La détection des spams en fait partie. Cette domaine (détection des spam) est très large qui permet aux scientifiques de travailler et de continuer à élargir de recherche.

Le domaine bio-inspiré qui est riche en méthodes et techniques qui ont montré leur efficacité dans différents domaines. Alors notre L'objectif du travail est d'utilisé une nouvelle méthode bio inspirée dans le domaine de détection du Spams (on parle de ASR et SWBs). Le domaine de détection de spam a particulièrement progressé ces dix dernières années, grâce à l'introduction des techniques héritées de l'apprentissage automatique qui ont amélioré significativement le taux du filtrage de spam, par la progression de classification des emails en spam et légitime. À l'heure actuelle, les techniques de filtrage de spam à base d'apprentissage sont loin d'être performants à 100%. Pour cela on a utilisé des nouvelles méthodes bio-inspiré cité sur notre étude qui a été portée sur la modélisation du processus de détection de spam par bio mimétisme en inspirants de la vie sociale des cafards et abeilles.

Notre application qui basée sur le système par les vies sociales des cafards et des abeilles on trouve à des déférents résultats. Nous avons obtenu des bons résultats. Néanmoins, les résultats obtenus par notre classificateur sont inférieurs à ceux obtenus par d'autres applications qui ont prouvé leurs efficacités pour bien extraire les entités nommées (algorithmes implémentés par le logiciel Weka).

Bibliographie :

- Abdiclie. F., et B. Atinani. « Extraction de Règles de Classification à partir des Données Spatiales. » Proceeding of the 2nd Conférence Internationale sur l'informatique et ses Applications (CIIA'09). Saida, Algeria. 2009.
- Agrawal, R., et R. Srikant. «Fast Algorithm for Mining Association Rules. » Proceedings of the 20th VLDB Conference. 1994.
- Androutsopoulos, I., G. Paliouras, V. Karkaletsis, G. Sakkis, et C. D. Spyropoulos. «Learning to filter spam e-mail: a comparison of a naïve Bayesian and a memorybased approach. » Proceedings of the Workshop on Machine Learning and Textual Information Access. Lyon, France. 2000b.
- Androutsopoulos, L, J. Koutsias, K. V. Chandrinos, et C. D. Spyropoulos. «An Evaluation of Naive Bayesian Networks. » In proceeding tsf of the Workshop on Machine Learning in the New Information Age. Barcelona, Spain, 2000a. pp. 9-17.
- Sanz E.P. et al, Email spam filtering, Advances in computers, vol.74, 2008, pp. 45-114.
- wikipedia, www.wikipedia.com, consulté le : 14/04/2020
- Arobase, www.arobase.org, consulté le : 28/04/2020
- aidewindows, www.aidewindows.net/phishing.php, consulté le : 28/04/2020
- sebsauvage, www.sebsauvage.net/comprendre/spam/index.html, consulté le : 28/04/2018
- Hassan, Algorithme de boosting et méta-heuristique basée sur la PSO Pour La détection et le Filtrage De Spam, Thèse de Master, Université Tahar Moulay-SAIDA, 2013
- statista, www.statista.com, consulté le : 29/04/2020
- S. Gastellier-prevost, Le spam, 2009. [10] G. Schryen, Anti-Spam Measures Analysis and Design, Berlin Heidel berg New York, Springer, 2010.
- Anti-spam, www.anti-spam.fr, consulté le : 02/05/2020
- Nouman Azam, Comparative Study of Features Space Reduction Techniques for Spam Detection, Thèse de Master, National University of Sciences & Technology, Pakistan.
- frameip ,www.frameip.com/spam-ham-antispam, consulté le : 04/05/2020
- M. S. El Bazzi, T. Zaki, D. Mammass, A. Ennaji, Indexation automatique des textes arabes : état de l'art, 2016. [15] B.S. Harish, D.S. Guru et Manjunath, Representation and Classification of Text Documents: A Brief Review, Recent Trends in Image Processing and Pattern Recognition, RTIPPR, 2010.

- @miscosman1996metaheuristics, title=Metaheuristics : A bibliography, author=Osman, Ibrahim H and Laporte, Gilbert, year=1996, publisher=Springer
- @articledeneche2006approches, title=Approches bio-inspirées pour la reconnaissance de formes, author=Deneche, Abdelhakim, year=2006, publisher=Université Mentouri Constantine
- @articlesaid2013methodes, title=Méthodes bio-inpirées hybrides pour la résolution de problèmes complexes, author=Said, Le journal= Université Constantine, volume=2, year=2013
- Bouarara, H. A., Hamou, R. M., & Amine, A. (2015). Novel Bio-Inspired Technique of Artificial Social Cockroaches (ASC). International Journal of Organizational and Collective Intelligence, 5(2), 47–79. Doi:10.4018/IJOCI.2015040103
- Hamou, R. M., Amine, A., & Boudia, A. (2013). A New Meta-Heuristic Based on Social Bees for Detection and Filtering of Spam. International Journal of Applied Metaheuristic Computing, 4(3), 15–33. Doi:10.4018/ ijamc.2013070102.
- La liste des mots vide français est disponible dans: http://www.ranks.nl/stopwords/french La liste des mots vide anglais est disponible dans:http://www.ranks.nl/stopwords
- [Harishetal.,2010] Harish,B..,Guru,D..,and Manjunath, S.(2010).Representation and classification of text documents: Abrief review, recent trends in image processing and pattern recognition. RTIPPR
- WEKA, www.cs.waikato.ac.nz/ml/weka/, consulté le : 12/05/2018
- Department of Telematics, www.dt.fee.unicamp.br/ ~ tiago/ sms spam collection, consulté le: 15/05/2020